INTERNATIONAL STANDARD

ISO/IEC 12034-1

First edition
2017-12

Information technology — Archive eXchange Format (AXF) —

Part 1: Structure and semantics

Technologies de l'information — Archive eXchange Format (AXF) — Partie 1: Structure et sémantique



Reference number
ISO/IEC 12034-1:2017(E)

© ISO/IEC 2017



COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2017, Published in Switzerland

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Ch. de Blandonnet 8 • CP 401
CH-1214 Vernier, Geneva, Switzerland
Tel. +41 22 749 01 11
Fax +41 22 749 09 47
copyright@iso.org
www.iso.org

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see the following URL: www.iso.org/iso/foreword.html.

ISO/IEC 12034-1:2017 was prepared by the Society of Motion Picture and Television Engineers (SMPTE) and was adopted, under the PAS procedure, by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, in parallel with its approval by the national bodies of ISO and IEC.

© ISO/IEC 2017 – All rights reserved

ISO/IEC 12034-1
SMPTE ST 2034-1:2017

ISO/IEC 12034-1 STANDARD

Archive eXchange Format (AXF) — Part 1: Structure & Semantics

Table of Contents

Page

Foreword 3
Intellectual Property 3
Introduction 3
1 Scope 7
2 Conformance Notation 7
3 Normative References 7
4 Definitions 8
5 Storage Media Types 13
5.1 Media with File Systems 14
5.2 Media without File Systems 14
5.3 File Marks 15
5.4 Relationships Between AXF Structures and Storage Media Types 15
6 Archive eXchange Format (AXF) Structure 16
6.1 Form of Data Expression 16
6.2 Byte Order 17
6.3 General AXF Concepts 17
6.4 AXF Data Structures 18
7 General Usage Considerations 36
7.1 File Naming 36
7.2 Media Preparation 36
7.3 AXF Object Index Structures 37
7.4 Creating, Reading, Writing, Copying, and Transferring AXF Objects 38
7.5 Nesting AXF Objects 39
8 Spanning 39
8.1 Spanning Linkages 39
8.2 Encountering a Spanning Situation 43
8.3 Recovery of Spanned AXF Objects 43
8.4 Spanning Rules 43
9 Collected Sets 44
9.1 Collected Set Linkages 44
9.2 Collected Set Structure 45
9.3 Add/Replace/Delete Processes 45
9.4 Tracking Versions 46

© ISO/IEC 2017 – All rights reserved

10 AXF Data Model 48
10.1 AXF Medium Identifier 49
10.2 Object Header 52
10.3 Object Fragment Header 58
10.4 File Footer 61
10.5 Object Fragment Footer 63
10.6 Object Footer 68
10.7 AXF Object Index 74
10.8 UUID 78
10.9 PositionInteger 78
10.10 FileFolder 79
10.11 Folder 81
10.12 File 84
10.13 Symlink 87
10.14 FileTree 90
10.15 Application 91
10.16 Entity 93
10.17 Location 95
10.18 Identifiers 97
10.19 Checksums 97
10.20 Identifier 97
10.21 Checksum 98
10.22 ByteOrder 100
10.23 Media Type 101
10.24 Structure Version 101

© ISO/IEC 2017 – All rights reserved

Foreword

SMPTE (the Society of Motion Picture and Television Engineers) is an internationally-recognized standard developing organization. Headquartered and incorporated in the United States of America, SMPTE has members in over 80 countries on six continents. SMPTE's Engineering Documents, including Standards, Recommended Practices, and Engineering Guidelines, are prepared by SMPTE's Technology Committees. Participation in these Committees is open to all with a bona fide interest in their work. SMPTE cooperates closely with other standards-developing organizations, including ISO, IEC and ITU.

SMPTE Engineering Documents are drafted in accordance with the rules given in its Standards Operations Manual.

SMPTE ST 2034-1 was prepared by Technology Committee 31FS on File Formats and Systems.

Intellectual Property

At the time of publication, no notice had been received by SMPTE claiming patent rights essential to the implementation of this Engineering Document. However, attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. SMPTE shall not be held responsible for identifying any or all such patent rights.

Introduction

This section is entirely informative and does not form an integral part of this Engineering Document.

The Archive eXchange Format (AXF) is an open format that supports interoperability among disparate data storage systems and ensures long–term availability of data, no matter how storage or file system technologies evolve. AXF inherently supports interoperability between existing, discrete storage systems, irrespective of the operating and file systems used, and also future–proofs digital storage by abstracting the underlying technology so that content remains available across generations of technology development.

At the most basic level, AXF is a file container that can encapsulate any number, size, and type of files in a fully self–contained and self–describing package. The package contains its own light-weight file system, which establishes independence from underlying operating systems, storage technologies, and file systems and can store any type of data on any type of storage media. Inside its packaging, AXF can contain metadata of any format, applicable to either AXF Objects or to individual files contained within AXF Objects; AXF also carries key preservation information, such as provenance, fixity, and the like — all key to ensuring long-term robustness and recoverability.

Historically, digital archive systems have used media data storage formats that are proprietary to their manufacturers, either intentionally or due to the lack of established standards. There have been neither interchange of media nor interoperability of archive systems between manufacturers and in some cases between different archive systems from the same manufacturer. Archives could be orphaned due to support ending for the systems used to create data archives. End users and manufacturers recognized that the proprietary nature of archive systems and the data stores that they create result in significant costs of operation that are unnecessary. These costs could be avoided if there were standardization of the format used for storage of the data on media and for transfer of the data between systems and locations. AXF permits separating the stored content from the systems that create and recover sets of data, thereby enabling refreshing of storage technology, recovering sets of data that otherwise would have been orphaned, and transferring sets of data between systems and locations.

This standard specifies a structure for data that can be written to any current or future data storage subsystem, regardless of the type of media on which it is stored. The data can include any types of files and associated metadata that are stored and transferred together in a structure called an "AXF Object." A single AXF Object can be spanned across multiple physical media, can be copied from one set of physical media to another, and

© ISO/IEC 2017 – All rights reserved

is agnostic to the Storage Media Type on which it is stored, e.g., spinning disc or linear tape. Regardless of the Storage Media Types on which they are stored, AXF Objects are identically structured and formatted for any given set and relationship of contained files and metadata.

AXF initially arose from the storage needs of the audiovisual production and archiving communities but quickly encompassed any type of file-based data. The transition to file-based workflows led to a new set of requirements throughout pre-production, production, distribution, storage, and preservation processes. Those requirements included long-term archiving of finished and unfinished materials, writing data to any type of storage subsystem using a standard scheme, transporting formatted archives between systems and locations using either media or networks, and allowing extensibility sufficient to accommodate any type of file, of any size, from any source, as well as adoption of any future storage technologies. AXF was created to address these requirements.

Audiovisual content archiving spans a wide range of content and data archiving systems and practices. At the time this standard was written, many different methods and media were commonly used to store file-based audiovisual content and its supporting information. Examples range from individual hard drives, solid state drives, and linear magnetic tape drives in small organizations to large spinning disc arrays in combination with very large robotic systems with multiple robots, each having multiple drives, in very large cultural, scientific, and legal archives. Applications in other industries that could benefit from the methods defined herein include medical imaging, geophysical exploration, scientific research, and similar high-volume producers of data.

The cultural, scientific, and business value of assets stored on these data systems is significant. Methods for storage, interchange, transport, and preservation of such assets, both locally and remotely, over both short and very long retention periods, demands a standardized, well-documented, non-manufacturer-specific method of writing data to any data storage system, from which the data then can be recovered and its contents used, updated, or transferred to another data storage system. All that would be necessary to achieve these objectives is a mechanism for recovering data from the media on which it is stored, plus utilities or applications that implement AXF.

The AXF standard creates a common method of writing individual files or related sets of files, and relevant metadata, onto data storage subsystems so that the structure of an AXF Object will remain the same no matter what vendor equipment or Storage Media Type is used. As long as the media remains viable and data can be read from that media, it will be possible to recover an AXF Object and unwrap its contents with a suitable utility or application running on whatever platform is current at the time. The AXF Object also has to be able to be recovered and stored on future data storage systems without requiring any changes to its contents simply to accomplish the act of medium migration, but it also needs to allow changes to its contents, in case updating is needed to data that already has been archived.

AXF addresses these needs through a combination of predefined eXtensible Markup Language (XML) schema fields, defined binary data structures that enable an AXF Object to carry any type of file within its File Payload, internal file system functionality, and key metadata enabling the spanning of AXF Objects across multiple physical media. The XML schema also enables essential information about an AXF Object and its contents to be read without having to process all the information within the AXF Object.

In addition to media interchange, AXF enables the interoperability of disparate systems through networks because it is structured as a streaming data set. Such interconnections enable seamless movement of AXF Objects from systems that create them, to systems that do not recognize the AXF protocol but store the AXF Object files nonetheless (perhaps in “cloud” storage), then to systems that are designed to recover data from AXF Objects.

Functionally, AXF acts like a file wrapper or a repository for all types of data without constraint. Unlike media-centric file formats such as MXF, which are similar in that they wrap essences, AXF can contain any number or types of files of any size encapsulated in an AXF Object. It is applicable across a much broader variety of file storage user groups than any media-specific file wrapper. Types of data can include media essence files, related metadata files, production files (such as word processing documents, hypertext documents, associated essence, applications, spreadsheets, and database copies), or any other type of data that users wish to store together. Unlike other file wrapper definitions, it is payload agnostic and does not require any special mappings or adaptations to accept the data an AXF Object carries.

AXF accommodates very large file sizes and quantities within AXF Objects. In the current version of this standard, 64-bit numbers are used to define the sizes of various parameters applicable to elements of AXF

© ISO/IEC 2017 – All rights reserved

Objects. 64-bit numbers can express values up to 18.44674 x 10^{18} (e.g., 18.44674 petabytes). Use of 64-bit numbers thus can define file sizes in bytes, numbers of files, numbers of media in a spanned set, and similar characteristics up to 18.44674 x 10^{18} of any particular element. If future requirements exceed the number spaces provided in this document, there is nothing fundamental that limits any particular parameter to expression using a 64-bit number. Future revisions of this standard could adopt larger number spaces (e.g., 96-bit, 128-bit, etc.) for those parameters requiring them. The net result is effectively unlimited storage capability within AXF Objects, in terms of file sizes, numbers of files in an AXF Object, number of AXF Objects on a medium, number of media in a spanned set, and the like.

AXF enables updating AXF Objects when additions of, modifications to, or deletions of files or information that they contain are needed. The functionality to modify AXF Objects is provided by linking “Supplemental” AXF Objects, written into an archive system at a later time to an original (“Anchor”) AXF Object. A Supplemental AXF Object updates contents of previous AXF Objects without requiring the original AXF Object itself to be modified. Since the original content of the AXF Object is retained in its original form, it is possible to restore either the original or the modified version whenever necessary. Additional Supplemental AXF Objects can be added in a chain, with restoration of the current or any earlier version possible at any time. When AXF Objects are refreshed by copying them to new media, it is possible to consolidate an Anchor Object and its Supplemental Object(s) into a single, new AXF Object. In doing so, it is possible to retain all of the constituent Objects of the Collected Set to which they belong, so that all earlier versions still can be reconstituted in the future.

AXF abstracts the storage of data from the applications that create AXF Objects and from the operating systems, file systems, drivers, and drives that store data on media. By this mechanism, any of the surrounding hardware and software components of systems can be replaced without affecting the data and its formatting within AXF Objects. A simplified view of where AXF fits into a basic stack is shown in Figure 1.

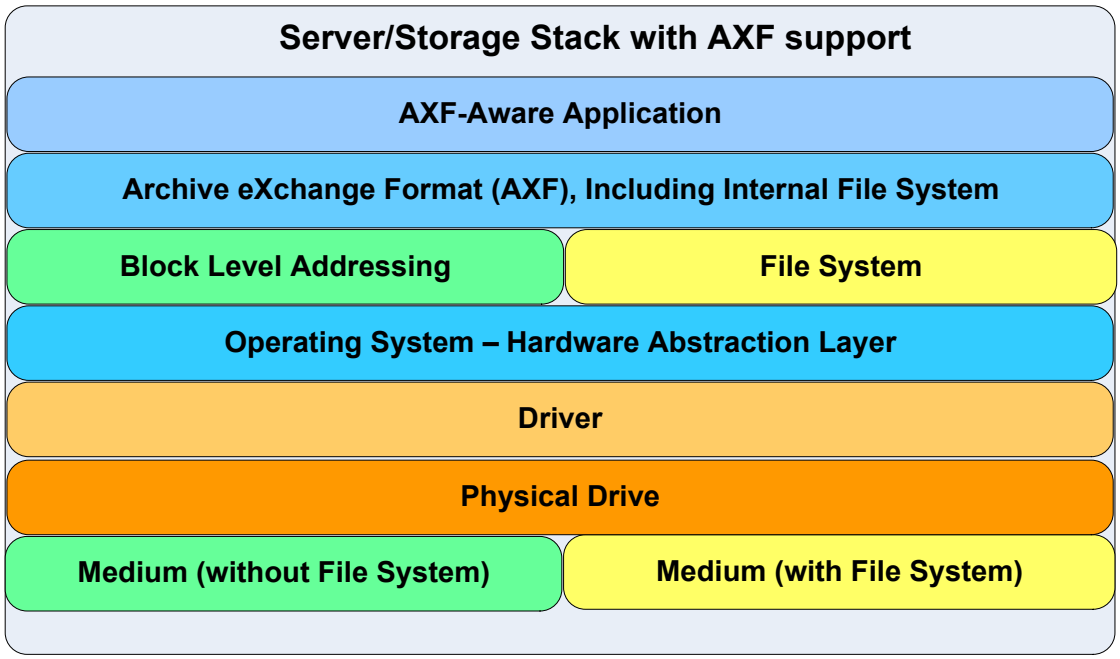


Figure 1 – Hardware/Software Stack Incorporating AXF Writing to and Reading from Media

AXF is designed so that each AXF Object comprises four main components, regardless of the technology that is used to store it. These components are:

Object Header – Each AXF Object begins with an Object Header, which contains descriptive XML metadata such as a unique identifier (UUID) for the AXF Object, information regarding its origin, its creation date, and a full index of all the files and folders contained in the Object, including file permissions and the like.

© ISO/IEC 2017 – All rights reserved

Generic Metadata Containers – Following an Object Header can be any number of optional Generic Metadata Containers. Such containers are self–contained, open metadata containers in which applications can place AXF-Object-specific metadata that is not part of the AXF Object File Payload. The metadata can be structured or unstructured, open or vendor-specific, binary, XML, plain text, or any other format.

AXF Object File Payload – Following any Generic Metadata Containers is the AXF Object File Payload. It contains the files encapsulated in the AXF Object. The File Payload consists of any number of triplets: File Data + File Padding + File Footer. File Padding ensures alignment of all AXF Object elements on the boundaries of Chunks into which each AXF Object is divided, thereby enabling addressing, by location within the AXF Object, by its internal file system. File Footer structures contain full information about the preceding file, along with a file-level checksum designed to be processed on-the-fly, with little or no overhead, during restore operations by an application. The information in File Footers enhances the resilience of AXF, as it can be used to recover File Payload data even if Object Header and Footer structures are missing or corrupt.

Object Footer – Completing an AXF Object is an Object Footer. It repeats the information contained in the Object Header and adds information captured during creation of the AXF Object, including per-file checksums, precise file sizes, and file positions within the AXF Object. The Object Footer is important to the interchange of an AXF Object because it allows efficient indexing by foreign systems when the media content is not previously known, thereby enabling media transport between systems that follow the AXF standard. It is one of the key structures that support the self-describing nature of AXF.

Other significant structures in the AXF protocol are AXF Medium Identifiers and AXF Object Indices. AXF Medium Identifiers are used on media to indicate formatting of the media according to the AXF protocol and to provide unique identification of the media. AXF Object Indices are optional compilations of the information in all Object Footers preceding each AXF Object Index on a medium, providing a single structure from which it is possible to obtain complete information on the contents of the preceding portion of a medium. When an AXF Object Index is the last structure on a medium, complete information about all AXF Objects stored on the medium can be obtained efficiently in one place.

AXF does not require a system to be fully compliant with this standard for it to be able to use and store AXF-generated AXF Objects. The initial adoption of AXF is anticipated to be in applications that create AXF Objects that then are stored on non-AXF-aware storage systems. Because the AXF Objects do not require a storage system to know that the AXF Objects are AXF-formatted, the AXF Objects will be viewed simply as files to be stored and retrieved. All that will be necessary to read AXF Objects will be the software and hardware needed to read the physical storage medium. As adoption grows, files can be moved into and out of AXF-aware systems as necessary, with the full range of features becoming available on systems that are AXF compliant. AXF-compliant applications will be able to read stored AXF Objects from any current operating system without unpacking entire AXF Objects to see critical metadata. Moreover, Archives, or AXF Objects within archives, also can span different types of media, allowing for flexibility within mixed-media archives and for AXF Objects to be identical, regardless of the media on which they are stored.

AXF offers resilience to data corruption and loss. AXF Object Indices, repeated identifier instances, and cryptographic hash checksums on both contents and AXF Objects allow for data corruption to be identified and mitigated. Even in catastrophic events, such as the loss of an external database containing records of the contents of an archive, the content database can be recreated by reading the archive and regenerating the archive-wide database from the records within the AXF Objects. This standard also enables the addition of more powerful data corruption recovery methods in future revisions by including provisions for incorporating forward error correction codes. AXF provides key features to identify and treat data corruption and loss across all types of storage formats. It allows big data and small to be processed in a consistent and standardized way.

Additional information on AXF can be found at OpenAXF.org

© ISO/IEC 2017 – All rights reserved

1 Scope

This standard is Part 1 of a series of documents that specify a general-purpose format for the storage and/or communication of information in bulk form. The format is named the Archive eXchange Format (AXF). The format described is intended both for interchange between systems and to serve as a native format within systems.

This standard identifies two major categories of data storage media and specifies the basic structures of data stored on those Storage Media Types. It specifies a number of structural elements for use in constructing the appropriate structures for use on each of the Storage Media Types. It defines the semantics of data contained within fields specified for use in the structural elements. The structural elements themselves are documents coded in the eXtensible Markup Language (XML), and this document defines an XML Schema Description (XSD) file for use in formulating the XML documents to be used for the structural elements of AXF Objects.

2 Conformance Notation

Normative text is text that describes elements of the design that are indispensable or text that contains the conformance language keywords: "shall," "should," or "may." Informative text is text that is potentially helpful to the user, but not indispensable, and that can be removed, changed, or added editorially without affecting interoperability. Informative text does not contain any conformance keywords.

All text in this document is, by default, normative, except: The Introduction, any section explicitly labeled as "Informative," or individual paragraphs that start with "Note:".

The keywords "shall" and "shall not" indicate requirements strictly to be followed in order to conform to the document and from which no deviation is permitted.

The keywords, "should" and "should not" indicate that, among several possibilities, one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required; or that (in the negative form) a certain possibility or course of action is deprecated but not prohibited.

The keywords "may" and "need not" indicate courses of action permissible within the limits of the document.

The keyword “reserved” indicates a provision that is not defined at this time, shall not be used, and may be defined in the future. The keyword “forbidden” indicates “reserved” and, in addition, indicates that the provision never will be defined in the future.

A conformant implementation according to this document is one that includes all mandatory provisions ("shall") and, if implemented, all recommended provisions ("should") as described. A conformant implementation need not implement optional provisions ("may") and need not implement them as described.

Unless otherwise specified, the order of precedence of the types of normative information in this document shall be as follows: Normative prose shall be the authoritative definition; tables shall be next, followed by formal languages, then figures, and then any other language forms.

3 Normative References

The following standards contain provisions that, through reference in this text, constitute provisions of this standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below.

© ISO/IEC 2017 – All rights reserved

XML 1.0, World Wide Web Consortium (W3C) (2008, November) Extensible Markup Language 1.0 (Fifth Edition)

XML Namespaces 1.0, World Wide Web Consortium (W3C) (8 December 2009) Namespaces in XML 1.0 (Third Edition)

XML Schema Part 1 — World Wide Web Consortium (W3C) (28 October 2004), XML Schema Part 1: Structures (Second Edition)

XML Schema Part 2 — World Wide Web Consortium (W3C) (28 October 2004), XML Schema Part 2: Datatypes (Second Edition)

ISO/IEC 10646:2012, Information Technology — Universal Coded Character Set (UCS), Third Edition (1 June 2012)

ISO/IEC 19505-1:2012, Object Management Group (OMG) (August 2011), Unified Modeling Language: Infrastructure, v2.4.1

ISO/IEC 19505-2:2012, Object Management Group (OMG) (August 2011), Unified Modeling Language: Superstructure, v2.4.1

IETF RFC 1494, Equivalence Between 1988 X.400 and RFC-822 Message Bodies (August 1993)

IETF RFC 2231, MIME Parameter Value and Encoded Word Extensions: Character Sets, Languages, and Continuations. (November 1997)

IETF RFC 3629, UTF-8, A Transformation Format of ISO 10646 (November 2003)

IETF RFC 3986, Uniform Resource Identifier (URI): Generic Syntax

IETF RFC 4122, A Universally Unique IDentifier (UUID) URN Namespace. (July 2005)

IETF RFC 6657, Update to MIME Regarding “Charset” Parameter Handling in Textual Media Types (July 2012)

IETF RFC 6838, Media Type Specifications and Registration Procedures

ISO/IEC 1001:2012(E) (VOL1), Information Technology — File Structure and Labeling of Magnetic Tapes for Information Interchange (2012-08-01)

ISO/IEC 13239:2012 (CRC), Information Technology — Telecommunications and Information Exchange Between Systems -- High-Level Data Link Control (HDLC) Procedures

FIPS PUB 180-4, Federal Information Processing Standards Publication, Secure Hash Standard *SHS), Information Technology Laboratory, National Institute of Standards and Technology, Gaithersburg, MD, March 2012

UTC, Coordinated Universal Time, ITU-R Recommendation TF 460-6, Standard-Frequency and Time-Signal Emissions, International Telecommunications Union, Radio Sector, Geneva, Switzerland, February 2002

4 Definitions

The following terms shall have the definitions specified for them below, both throughout this document and in related Parts of this SMPTE standard.

© ISO/IEC 2017 – All rights reserved

4.1 Anchor Object

The first AXF Object in a Collected Set of AXF Objects, the **SetSequence** value of which is 1. It is the AXF Object from which all Subsequent Objects in the Collected Set differentiate the contents of the Product Object.

4.2 Archive Object

An AXF Object.

4.3 Attribute

An XML markup construct consisting of a name/value pair that describes an element.

4.4 AXF

Archive eXchange Format.

4.5 AXF-Aware

Capable of reading, interpreting, and/or writing AXF Objects and other AXF data structures.

4.6 AXF Media

Multiple media prepared to carry AXF Objects.

4.7 AXF Medium

A medium prepared to carry AXF Objects.

4.8 AXF Medium Identifier

A data structure at the start or root of a medium that indicates that it has been prepared for storage of AXF Objects and that provides information about the identification, formatting, and history of the medium.

4.9 AXF Object

A group of data elements that are stored as a unit and that are contained within a single data structure.

4.10 AXF Object Index

A data structure, stored on a medium, that is a collection of the complete information about all previously-stored AXF Objects on that medium and the contents of those AXF Objects, as well as the locations of the content files stored within those AXF Objects.

4.11 Base Name

The fundamental name of a file; the portion of a file name prior to any file name extension that identifies the type of file or structure.

4.12 Big Endian

The order in which the bytes of a multi-byte number are transmitted with the most significant byte first.

4.13 Binary Structure Container

A data structure that is formed using easily identifiable binary data patterns of UTF-8 characters and that carries a payload of a specified type for the purpose of providing structural elements for AXF Objects and for the files contained within them.

4.14 Binary Structure Container Payload

The data contained within the **Payload** field of a Binary Structure Container.

4.15 Block

© ISO/IEC 2017 – All rights reserved

A section of data of uniform size (measured in bytes) as stored on a physical medium that is independently addressable for write and read operations, and which is defined as a “block” on certain Storage Media Types and as a “sector,” a “page” or some other term on other Storage Media Types.

4.16 Block-Based

Media that enable direct access to Blocks of data stored thereon and that do not incorporate or depend upon File Systems.

4.17 Block Position

The location on a medium of a block of data, measured as the number of blocks from the beginning of the medium, as counted by the drive on which the medium is carried.

4.18 Block Size

A parameter that defines the size of Blocks contained on a medium. Some Storage Media Types can accommodate varying block sizes across the storage space of a medium.

4.19 Byte Order Mark

A small sequence of bytes carried within a file from which it is possible to determine the byte ordering (Big Endian or Little Endian) of the contents of the file. (Abbreviated BOM.)

4.20 Chunk

A section of data of uniform size (measured in bytes) within an AXF Object.

4.21 Chunk Size

A parameter that defines the size of all of the Chunks within an AXF Object.

4.22 Collected Set

A group of AXF Objects, consisting of an Anchor Object and one or more Subsequent Objects, that, when compiled, produce a Product Object differing from the Anchor Object as a result of additions, replacements, or deletions of files.

4.23 Content File

A File of any type that is stored as part of the File Payload within an AXF Object. Also a Payload File.

4.24 Data Element

A file or data structure that forms part of the overall data stored within an AXF Object or on a medium.

4.25 Element

An XML logical document component bounded by matching start and end tags and including any content between the tags plus any associated attributes.

4.26 Embedded File System

A light weight File System having its database contained within an AXF Object.

4.27 Endianness

The ordering of bytes within a multi-byte number. Byte orders typically are Little Endian or Big Endian, although other, less frequently used byte orders exist.

4.28 File Mark

A data structure written or read by a Linear Media drive to provide separation between data elements stored on a medium using the media technology for which the drive is designed.

© ISO/IEC 2017 – All rights reserved

4.29 File Footer

A Binary Structure Container that carries a payload with information to identify the file that it follows and to provide detailed information about that file.

4.30 File Padding

Data inserted at the end of a file to completely fill the last Chunk occupied by the file.

4.31 File Payload

The portion of an AXF Object that carries the Payload Files and their associated File Footer Binary Structure Containers.

4.32 File-System-Based

Media that incorporate and depend upon File Systems and provide no direct access to Blocks of data.

4.33 File System

A type of data store that can be used to store, retrieve, and update a set of files.

4.34 File Tree

A hierarchical structure of folders and files, expressed in XML, that starts at a root point and establishes relationships between folders and the folders and files that they contain.

4.35 Fixity

Fixity is the property of being constant, steady, and stable. Fixity checking is the process of verifying that a digital object has not been altered or corrupted. In practice, this is most often accomplished by computing and comparing cryptographic hash checksums.

4.36 Fragment

A data structure that contains a portion of an AXF Object that is linked to preceding or following Fragments, or both, through shared UUID values.

4.37 Generic Metadata Container

A Binary Structure Container the payload of which is metadata.

4.38 Linear Media

Media requiring writing and reading sequentially along the length of the medium (typically tape).

4.39 Little Endian

The order in which the bytes of a multi-byte number are transmitted with the least significant byte first. This might or might not match the order in which numbers are normally stored in memory for a particular processor.

4.40 Media

Plural of medium.

4.41 Media Class

The general category of Media, that is, Block-Based Media (i.e., both Linear and Non-Linear Media without File System support) and File-System-Based Media (i.e., both Linear and Non-Linear Media with File System support).

4.42 Media Family

A group of related Storage Media Types sharing a common set of media technology but allowing for different implementation characteristics, such as with respect to storage density.

4.43 Media Technology

© ISO/IEC 2017 – All rights reserved

A set of techniques used to store data on a particular form of media, including the choice of medium itself and the methods for such functions as recording bits on a medium, error correction, indicating a separation between portions of data stored, and the like.

4.44 Media Type

A registered name that describes the format of the content of a file.

4.45 Medium

A physical carrier on which data is stored, which comprises an integer number of Blocks and which can be either Block-Based or File-System-Based. (Equivalent to **Storage Medium**.)

4.46 Non-Linear Media

Media that can be written, read, and accessed in a non-sequential manner, enabling data to be accessed randomly on such a medium.

4.47 Object Footer

A data structure at the end of an AXF Object that provides complete information about the contents of the AXF Object, the positions within the AXF Object of the files that it contains, and other information about the AXF Object and its contents.

4.48 Object Fragment

The portion of an AXF Object stored on a single medium when the AXF Object is spanned across multiple media.

4.49 Object Header

A data structure at the start of an AXF Object that provides some or all of the information about the contents of the AXF Object, the positions within the AXF Object of the files that it contains, and other information related to the AXF Object and its contents.

4.50 Padding

Data having a value of 0x00 used to fill data space to maintain the format of a data structure.

4.51 Padding Chunk

A Chunk completely filled with bytes having a value of 0x00.

4.52 Payload File

A file contained within an AXF Object following the Payload Start Binary Structure Container and preceding the Payload Stop Binary Structure Container, if present.

4.53 Product Object

The compilation of files, directory structure, and other information that results from the combination of the Anchor Object and all subsequent AXF Objects in a Collected Set.

4.54 Spanned Set

A group of media containing Fragments of a single AXF Object linked together by a shared UUID identifying the AXF Object and by shared UUID values matching the Span Points of the AXF Object.

4.55 Spanning

A process for fragmenting a single AXF Object and storing its fragments on multiple media.

4.56 Span Point

© ISO/IEC 2017 – All rights reserved

The location within an AXF Object at which the AXF Object contents are fragmented for storage in separate Fragments.

4.57 Storage Media Type

A specific combination of media technology with parameters for the various data storage characteristics and data processing methods.

4.58 Storage Medium

A Medium used to store data. (Equivalent to **Medium**.)

4.59 Subsequent Object

An AXF Object in a Collected Set of AXF Objects other than the Anchor Object, the **SetSequence** value of which is greater than 1. It is an AXF Object that differentiates the contents of the Product Object from the Anchor Object of the Collected Set by indicating additions, replacements, or deletions of files and that carries any files necessary to effectuate the additions or replacements of files that it indicates.

4.60 Symbolic Link

A pointer contained within a directory structure to a file or folder that may be internal or external to the file hierarchy described by the directory structure. Symbolic Links are used with certain types of File Systems, while other types of File Systems use “shortcut” files referenced by the directory structure to perform the pointer function.

4.61 Symlink

Symbolic Link

4.62 URI

Uniform Resource Identifier – A compact sequence of characters that identifies an abstract or physical resource.

4.63 UUID

Universally Unique Identifier – An identifier that is unique for all practical purposes and is created according to an algorithm specified to lead to different values for each such identifier generated.

4.64 VOL1

A label applied at the beginning of data tape media according to the provisions of ISO/IEC 1001 “Information technology – File structure and labeling of magnetic tapes for information interchange”

5 Storage Media Types

Media that carry AXF Objects in conformance with this standard fall into one of two fundamental categories: Block-Based and File-System-Based. The formatting of each of these Storage Media Types is according to specific arrangements described generally in this section and in greater detail below in other sections. As also described below, AXF Objects stored on the respective Storage Media Types do not differ in their basic formats, but certain indexes and pointers do differ somewhat in the values they carry depending upon the Storage Media Types upon which their AXF Objects are stored. Mechanisms are provided herein and in other Parts of this document suite for the storage, transfer and processing of AXF Objects between Storage Media Types in a transparent manner and using minimal processing. Block-Based and File-System-Based Media are either Linear or Non-Linear.

Linear Media are media that only can be written and read sequentially, from end to end. In the drives that write and read them, Linear Storage Media Types include mechanisms for writing and reading File Marks or similar identifiers of points of separation between sequential sections of data stored on the media. Linear Storage Media Types include in the drives that write and read them at least the capability to search for File Marks, to count File Marks over the length of a medium, and to advance in either direction by a specified number of File

© ISO/IEC 2017 – All rights reserved