INTERNATIONAL STANDARD

ISO/IEC 13346-4

First edition
1995-12-15

Information technology — Volume and file structure of write-once and rewritable media using non-sequential recording for information interchange —

Part 4:
File structure

Technologies de l'information — Structure de volume et de fichier de moyens d'écriture unique et de réécriture utilisant un enregistrement non séquentiel pour l'échange d'information —

Partie 4: Structure de fichier

Reference number
ISO/IEC 13346-4:1995(E)

Contents ..... Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialised system for worldwide standardisation. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organisation to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organisations, governmental and non-governmental, in liaison with ISO and IEC, also take part in this work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication of an International Standard requires approval by at least 75% of the national bodies casting a vote.

International Standard ISO/IEC 13346 was prepared by the European Association for Standardizing Information and Communication Systems, ECMA, (as Standard ECMA-167) and was adopted, under a special "fast-track procedure", by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, in parallel with its approval by National Bodies of ISO and IEC.

Annex A forms an integral part of this Part of ISO/IEC 13346.

Introduction

ISO/IEC 13346 is a volume and file structure standard for interchanging files and as such, it is a peer to existing volume and file structure standards such as ISO 9293 and ISO 9660. It is rather different from those standards in at least two important ways. Firstly, it offers much more functionality, mainly because of user needs for increased character set support and for more powerful file system features. Secondly, it acknowledges the separate concerns of booting, volume structure and file system structure. Rather than bundling these different functions together, ISO/IEC 13346 carefully segregates these functions into separate parts and describes in detail how those parts fit together. It is expected that future volume and file structure standards will fit into this framework, rather than building other distinct and incompatible formats.

ISO/IEC 13346 is published in five Parts. Part 1 - general - specifies references, definitions, notations and basic structures used in the other four Parts. Part 2 - volume and boot block recognition - specifies formats and system requirements for recognising the volume structures on a medium and booting from a medium. Part 3 - volume structure - specifies how to record various volume-related entities such as volumes, volume sets and logical volumes. Part 4 - file structure - specifies how to record and interpret files, both file data and file attributes, and file hierarchies within logical volumes. Part 5 - record structure - specifies how to record and interpret file data encoded as records.

Information technology - Volume and file structure of write-once and rewritable media using non-sequential recording for information interchange -

Part 4:
File structure

Section 1: General

1 Scope

ISO/IEC 13346 specifies a format and associated system requirements for volume and boot block recognition, volume structure, file structure and record structure for the interchange of information on media between users of information processing systems.

The media shall be recorded as if the recording of sectors may be done in any order.

NOTE 1 - The medium is not restricted to being of only one type; the type of medium may be either write once, or read only, or rewritable, or a combination of these types.

ISO/IEC 13346 consists of the following five Parts:

Part 1: General

Part 2: Volume and Boot Block Recognition

Part 3: Volume Structure

Part 4: File Structure

Part 5: Record Structure

Annex A - ICB Strategies, is part of this Part of ISO/IEC 13346.

This Part of ISO/IEC 13346 specifies a format and associated system requirements for file structure by specifying:

- the placement of files;
- the attributes of the files;
- the relationship among files of a logical volume;
- levels of medium interchange;
- requirements for the processes which are provided within information processing systems, to enable information to be interchanged between different systems; for this purpose, it specifies the functions to be provided within systems which are intended to originate or receive media which conform to this Part of ISO/IEC 13346.

2 Parts references

The first digit of a reference within ISO/IEC 13346 identifies the Part, e.g. 2/5 refers to clause 5 in ISO/IEC 13346-2, and figure 4/3 refers to figure 3 in this Part of ISO/IEC 13346.

3 Part interface

This clause specifies the interface of this Part of ISO/IEC 13346 to other standards or Parts.

3.1 Input

This Part of ISO/IEC 13346 requires the specification of the following by another standard or Part.

- Volume sets of one or more volumes.
- A means for assigning volume sequence numbers (see 4/8.1).
- Logical volumes composed of partitions.

- Numeric identification of the partitions within a logical volume.
- If the volume is recorded according to ISO/IEC 13346-3, the partitions shall be numbered according to 3/8.8.
- Identification of a logical volume on which one or more file sets may be recorded.
- Division of the partitions for a logical volume into fixed size logical blocks.
- Numeric identification of the logical blocks within a partition.
- The size of a logical block for a logical volume. This shall be an integral multiple of 512.
- A means for detecting if a logical block is unrecorded.
- If the volume is recorded according to ISO/IEC 13346-3, a logical block shall be unrecorded if all of the logical sectors comprising that logical block are unrecorded. A logical block should either be completely recorded or unrecorded.
- A means for identifying the first extent of the File Set Descriptor Sequence (see 4/8.3.1) of a logical volume;
- If the volume is recorded according to ISO/IEC 13346-3, the extent in which the first File Set Descriptor Sequence of the logical volume is recorded shall be identified by a long_ad (4/14.14.2) recorded in the Logical Volume Contents Use field (see 3/10.6.7) of the Logical Volume Descriptor describing the logical volume in which the File Set Descriptors are recorded.
- A means for specifying the Logical Volume Header Descriptor (see 4/14.15) of a logical volume
- If the volume is recorded according to ISO/IEC 13346-3, the Logical Volume Header descriptor shall be recorded in the Logical Volume Contents Use field (see 3/10.10.5) of the prevailing Logical Volume Integrity Descriptor for the logical volume.
- A means for identifying the following for each partition of the logical volume on which a file set is recorded:
  - Unallocated Space Table and Unallocated Bit Map (see 4/10)
  - Freed Space Table and Freed Bit Map (see 4/10)
  - Partition Integrity Table (see 4/11)

If the volume is recorded according to ISO/IEC 13346-3, the Partition Contents Use field (see 3/10.5.6) of the Partition Descriptor (see 3/10.5) describing the partition shall be recorded as a Partition Header Descriptor (see 4/14.3). Such a Partition Descriptor shall have "+NSR02" recorded in the Partition Contents field.

3.2 Output

This Part of ISO/IEC 13346 specifies the following which may be used by other standards or Parts.

- Data space of a file (see 4/8.8.2).
- Attributes of a file.
- Attributes of a directory.
- Attributes of a directory hierarchy.

4 Conformance

4.1 Conformance of a medium

A medium shall be in conformance with ISO/IEC 13346 when it conforms to a standard for recording (see 1/5.10) and information recorded on sectors of the medium conform to the specifications of ISO/IEC 13346-1 and one or more of Parts 2, 3, 4 and 5. A statement of conformance shall identify the sectors of the medium on which information is recorded according to the specifications of ISO/IEC 13346, and the Parts and the levels of medium interchange (see 2/10, 3/11, and 4/15) to which the contents of those sectors of the medium conform.

4.2 Conformance of an information processing system

An information processing system shall be in conformance with ISO/IEC 13346 if it meets the requirements specified in ISO/IEC 13346-1 and one or more of Parts 2, 3, 4 and 5 either for an originating system (see 2/12, 3/13, 4/17 and 5/11) or for

a receiving system (see 2/13, 3/14, 4/18 and 5/12) or for both types of system. A statement of conformance shall identify the Parts, and the levels of the requirements for each of those Parts, which can be met by the system.

5 Definitions

For the purposes of this Part of ISO/IEC 13346, the definitions given in ISO/IEC 13346-1 (see 1/5) and the following definitions apply.

5.1 **extent:** A set of logical blocks, the logical block numbers of which form a continuous ascending sequence. The address, or location, of an extent is the number of the first logical block in the sequence.

5.2 **file set:** A collection of files and directories.

5.3 **group ID:** An identification of a group of users.

5.4 **logical block:** The unit of allocation of a logical volume.

5.5 **logical volume:** A nonempty set of partitions over which one or more file sets are recorded.

5.6 **partition:** An extent of logical blocks within a volume.

5.7 **user ID:** An identification of a user.

6 Notation

The notation of ISO/IEC 13346-1 (see 1/7) applies to this Part of ISO/IEC 13346.

7 Basic types

In addition to the basic types of ISO/IEC 13346-1 (see 1/7), the following basic types apply for this Part of ISO/IEC 13346.

7.1 Recorded address

A logical block address may be specified by an `lb_addr` recorded in the format shown in figure 4/1.

RBP	Length	Name	Contents
0	4	Logical Block Number	`Uint32` (1/7.1.5)
4	2	Partition Reference Number	`Uint16` (1/7.1.3)

Figure 1 - `lb_addr` format

7.1.1 Logical Block Number (RBP 0)

This field specifies the logical block number relative to the start of the partition identified by the Partition Reference Number field. The value 0 shall refer to the first logical block in the partition.

7.1.2 Partition Reference Number (RBP 4)

This field contains the numeric identification of a partition within a logical volume (see 4/3.1).

7.2 Descriptor Tag

Certain descriptors specified in this Part of ISO/IEC 13346 have a 16 byte structure, or `tag`, at the start of the descriptor with the format given in figure 4/2.

NOTE 2 - There are two main motivations for using a general tag structure. The first is that most descriptors need a generic way to handle issues of CRCs and format versions. The second motivation is to support recovery after the medium has been damaged or corrupted in some (unspecified) way. With the tag described here, structures are self identifying and can be verified with very little context.

RBP	Length	Name	Contents
0	2	Tag Identifier	`Uint16` (1/7.1.3)
2	2	Descriptor Version	`Uint16` (1/7.1.3)
4	1	Tag Checksum	`Uint8` (1/7.1.1)
5	1	Reserved	#00 byte
6	2	Tag Serial Number	`Uint16` (1/7.1.3)
8	2	Descriptor CRC	`Uint16` (1/7.1.3)
10	2	Descriptor CRC Length	`Uint16` (1/7.1.3)
12	4	Tag Location	`Uint32` (1/7.1.5)

Figure 2 - `tag` format

7.2.1 Tag Identifier (RBP 0)

This field shall specify an identification of the descriptor type. Type 0 shall specify that the format of this descriptor is not specified by this Part of ISO/IEC 13346. Types 1-7 and 9 are specified in Part 3. Type 8 is specified identically in Part 3 and this Part of ISO/IEC 13346. Types 256-265 are specified in this Part of ISO/IEC 13346. All other types are reserved for future standardisation. The descriptor types specified by this Part of ISO/IEC 13346 are shown in figure 4/3.

Type	Interpretation
8	Terminating Descriptor (3/10.9 and 4/14.2)
256	File Set Descriptor (4/14.1)
257	File Identifier Descriptor (4/14.4)
258	Allocation Extent Descriptor (4/14.5)
259	Indirect Entry (4/14.7)
260	Terminal Entry (4/14.8)
261	File Entry (4/14.9)
262	Extended Attribute Header Descriptor (4/14.10.1)
263	Unallocated Space Entry (4/14.11)
264	Space Bitmap Descriptor (4/14.12)
265	Partition Integrity Entry (4/14.13)

Figure 3 - Descriptor interpretation

7.2.2 Descriptor Version (RBP 2)

This field shall specify the version of this descriptor. The value 2 shall indicate the structure of this Part of ISO/IEC 13346.

7.2.3 Tag Checksum (RBP 4)

This field shall specify the sum modulo 256 of bytes 0-3 and 5-15 of the `tag`.

7.2.4 Reserved (RBP 5)

This field shall be reserved for future standardisation and shall be set to #00.

7.2.5 Tag Serial Number (RBP 6)

This field shall specify an identification of a set of descriptors. If the field contains 0, then no such identification is specified.

NOTE 3 - This field can be used to distinguish between groups of descriptors. For example, when reusing rewritable media, an implementation might choose a different serial number from the previous use when initialising a volume. Thus, a disaster recovery mechanism can avoid recovering prior and unintended data. The only alternative to this scheme would be to force volume initialisation to clear the volume.

7.2.6 Descriptor CRC (RBP 8)

This field shall specify the CRC of the bytes of the descriptor starting at the first byte after the descriptor tag. The number of bytes shall be specified by the Descriptor CRC Length field. The CRC shall be 16 bits long and be generated by the CRC-ITU-T polynomial (see ITU-T V.41):

$x^{16} + x^{12} + x^5 + 1$

NOTE 4 - As an example, the CRC of the three bytes #70 #6A #77 is #3299. Implementations can avoid calculating the CRC by setting the Descriptor CRC Length to 0, as then the Descriptor CRC shall be 0.

7.2.7 Descriptor CRC Length (RBP 10)

This field specifies how many bytes were used in calculating the Descriptor CRC.

7.2.8 Tag Location (RBP 12)

This field shall specify the number of the logical block, within the partition the descriptor is recorded on, containing the first byte of the descriptor.

NOTE 5 - The location of the tag may appear to be redundant but its primary purpose is to make it extremely likely that if the first 16 bytes of a logical sector or logical block is a consistent descriptor tag, then it is a descriptor tag.

Section 2: Requirements for the medium for file structure

8 File structure

8.1 Volume set

Each volume in a volume set shall have an assigned volume sequence number as specified in 4/3.1.

8.2 Arrangement of information on a volume set

A logical volume and its related file sets shall be recorded on a volume set. Identification of the logical volumes (see 4/3.1) and related File Set Descriptors for all the logical volumes in a volume set shall be recorded in the volume with the highest volume sequence number in the volume set.

8.3 Arrangement of information on a logical volume

This Part of ISO/IEC 13346 takes a logical volume to be a set of partitions on one or more volumes. Each partition shall be considered as an extent of logical blocks, and shall have an identification as specified in the input parameters for this Part of ISO/IEC 13346 (see 4/3.1). An address within a logical volume has two parts; one part identifies a partition within the logical volume and the other part specifies a logical block number relative to the start of that partition.

8.3.1 File Set Descriptor Sequence

A File Set Descriptor Sequence shall be recorded as a sequence of extents within a logical volume. An extent of the File Set Descriptor Sequence shall be recorded according to the schema shown in figure 4/4.

```
[File Set Descriptor Sequence extent]{
        <File Set Descriptor>0+
  [Terminator]{
      <File Set Descriptor>
    | <Terminating Descriptor>
    | <unrecorded logical block>
  } <trailing logical block>0+
} 0+
```

Figure 4 - File set descriptor sequence schema

The first extent of the sequence shall be identified by the input parameters (see 4/3.1) of this Part of ISO/IEC 13346. Each, if any, subsequent extent of the sequence shall be identified by the Next Extent field of a File Set Descriptor. An extent of the sequence shall be terminated by either an unrecorded logical block (see 4/3.1), a Terminating Descriptor (see 4/14.2), or by a File Set Descriptor whose Next Extent field identifies a subsequent extent of the sequence.

All File Set Descriptors shall have an assigned file set descriptor number. All File Set Descriptors with identical file set descriptor numbers shall have identical contents.

All file sets shall have an assigned file set number. Of the File Set Descriptors of a File Set Descriptor Sequence with identical file set numbers, the one with the highest file set descriptor number shall be used. This instance shall be referred to as the prevailing instance.

One of the File Set Descriptors of a File Set Descriptor Sequence shall have a file set number of 0.

A File Set Descriptor shall specify a file set identification. No prevailing instance of a File Set Descriptor shall specify the same file set identification as any other prevailing instance of a File Set Descriptor.

8.4 Arrangement of information on a partition

A means for identifying the location of the following for each partition of the logical volume on which a file set is recorded shall be specified by the input parameters (see 4/3.1) of this Part of ISO/IEC 13346.

- Unallocated Space Table and Unallocated Bit Map (see 4/10)
- Freed Space Table and Freed Bit Map (see 4/10)
- Partition Integrity Table (see 4/11)

8.5 File set

A file set shall be identified by a File Set Descriptor which identifies the root of a directory hierarchy (see 4/8.6) describing a set of files and certain attributes of the file set. A prevailing File Set Descriptor specifies

- the name of the logical volume it is recorded on
- the set of characters allowed in certain fields of descriptors associated with the file set identified by the File Set Descriptor
- an identification of the root of the directory hierarchy describing the files of the file set identified by the File Set Descriptor
- copyright and abstract information for the file set.

8.6 Directories

A directory contains zero or more file or directory identifications. A directory hierarchy shall be a set of directories descended from a single root directory.

A directory shall contain a set of directory descriptors, each of which identifies a parent directory or a component file or a component subdirectory. A directory descriptor that identifies a parent directory or a component file or subdirectory by specifying the address of an ICB (see 4/8.10.1) for that component shall be recorded as a File Identifier Descriptor (see 4/14.4). A directory descriptor that identifies a component file or subdirectory of the directory by specifying the pathname of the actual file or directory shall be referred to as an alias and shall be recorded as a File Identifier Descriptor specifying a file whose type is a symbolic link (see 4/14.6.6).

A directory identifying another directory by other than an alias shall be called a parent directory of the identified directory. The identified directory shall be called a subdirectory of the parent directory. Different directories may have the same parent directory. A directory shall have only one parent directory. The parent directory of the root directory shall be the root directory.

Each directory descriptor shall specify the name of a component file or the name of a component subdirectory, or identify the parent directory of the directory. The length, in bytes, of the name of a component file or subdirectory shall be greater than 0. Each directory descriptor shall contain an indication of whether the identified component is a directory. When the descriptor identifies an alias, this indication is contained in the directory descriptor for the file or directory identified by the pathname specified by the alias.

A directory shall be recorded according to the schema shown in figure 4/5.

```
{
        <File Identifier Descriptor>
} 0+
```

Figure 5 - Directory schema

For the descriptors in a directory

- there shall not be more than one descriptor with the same File Identifier (see 4/14.4.8) and File Version Number (see 4/14.4.2).
- a descriptor identifying a directory shall have a File Version Number of 1.
- there shall be exactly one File Identifier Descriptor identifying the parent directory (see 4/14.4.3).
- the descriptors shall be ordered according to 4/8.6.1 and 4/14.6.8.

A File Entry specifying a file in which a directory is recorded shall not specify a Character Set Information Extended Attribute.

NOTE 6 - The character set specifying the d-characters (1/7.2) used in the directory descriptors is specified in the File Set Descriptor for the directory hierarchy of which the directory is a member.

8.6.1 Order of directory descriptors

If the directory descriptors of a directory are sorted according to this Part of ISO/IEC 13346, they shall be ordered by the following criteria in descending order of significance:

1. In ascending order according to the relative value of File Identifier, where File Identifiers shall be valued as follows:
   - If the File Identifiers being compared have the same value in all byte positions, the File Identifiers shall be equal in value.
   - If the File Identifiers being compared do not contain the same number of byte positions, the File Identifiers shall be treated as if they are of equal length by padding the shorter File Identifier on the right with #00 bytes. After any such padding, the File Identifiers shall be compared one byte at a time, in ascending byte position order, until a byte position is found that does not contain the same value in both File Identifiers. The File Identifier with the greater byte value, comparing values of the bytes as unsigned integers, shall be considered greater.
2. In descending order according to the relative value of File Version Number.

NOTE 7 - Sorting applies to files and aliases having been marked as deleted in the File Characteristics field. The order is independent of the `charspec` (1/7.2.1) applying to the directory because the File Identifiers are sorted as if they were binary values.

8.6.2 Directory hierarchy size restrictions

The sum of the number of directories and the number of files described by the directories of a directory hierarchy shall be less than 2^{32}.

8.7 Pathname

A pathname may be used to specify a file or directory by name. The length, in bytes, of this pathname shall be greater than 0. The pathname shall consist of a sequence of one or more path components (see 4/14.16) as follows:

- Unless otherwise specified, a component shall be interpreted relative to the directory specified by its predecessor. The predecessor of the initial component shall be the directory in which the pathname is described.
- The final component shall specify either a directory, or a file, or an alias which resolves to either a directory or file.
- Each other component shall specify either a directory or an alias which resolves to a directory.

8.7.1 Resolved pathname

Within a directory hierarchy, every pathname specifying a file or directory has an equivalent resolved pathname. A resolved pathname is a pathname where

- the first component is a Path Component with a Component Type of 2.
- each other component is a Path Component with a Component Type of 5 and is not an alias.

The length of a resolved pathname shall be the sum of the following:

- the value of the Component Identifier Length field for each component;
- the number of components

NOTE 8 - The resolved pathname is mainly used in 4/15. Note that the length defined here corresponds to that of a theoretical pathname, rather than a pathname that an implementation might use. In particular, it assumes that the component separator is one byte long, which is typically true but is false for certain character sets.

Note that the length of the resolved pathname does not provide for the length of the file version number associated with the final component of the pathname.

8.8 Files

A file shall be described by a File Entry (see 4/14.9) which shall specify the attributes of the file and the location of the file's recorded data. The data of a file shall be recorded in either of the following:

- An ordered sequence of extents of logical blocks (see `short_ad` (4/14.14.1), long_ad (4/14.14.2) and ext_ad (4/14.14.3). The extents may be recorded or unrecorded, and allocated or unallocated. The extents, if specified as long_ad (4/14.14.2) or ext_ad (4/14.14.3), may be located on different partitions which may be on different volumes.
- The Allocation Descriptors field of a File Entry.

Except where specified in this Part of ISO/IEC 13346, neither the interpretation of the information in a file nor the structure of the information in a file is specified by this Part of ISO/IEC 13346.

8.8.1 Attributes of a file

The File Entry specifies the attributes of a file. Some of the attributes shall be recorded in fields in the File Entry itself; the remainder shall be recorded as extended attributes. Extended attributes shall be recorded in extended attributes spaces as described in 4/9. The attributes of a file specified by this Part are recorded in extended attributes, and in the following fields of a File Entry and of the `icbtag` (4/14.6) recorded as the contents of the ICB Tag field of the File Entry.

`icbtag`

- Strategy Type
- Strategy Parameter
- File Type
- Flags

File Entry

- Uid
- Gid
- Permissions
- File Link Count
- Record Format
- Record Display Attributes
- Record Length
- Information Length
- Logical Blocks Recorded
- Access Date and Time
- Modification Date and Time
- Attribute Date and Time
- Checkpoint
- Implementation Identifier

NOTE 9 - The information in the File Identifier Descriptor (see 4/14.4.3) for a file pertains only to the identification of the file and is not considered an attribute of the file.

8.8.2 Data space of a file

The data space of a file shall be the following:

- If the file is recorded as the contents of an ordered sequence of extents of logical blocks, these extents shall be the data space of the file. The bytes in the data space shall be numbered with consecutive integers assigned in an ascending sequence. The numbering shall start from 0 which shall be assigned to the first byte of the first logical block of the first extent, if any, of the data space.
- If the file is recorded in the Allocation Descriptors field of a File Entry, the number of bytes specified by the Length of Allocation Descriptors field of the File Entry starting with the first byte of the Allocation Descriptors field of the File Entry shall be the data space of the file. The bytes in the data space shall be numbered with consecutive integers assigned in an ascending sequence. The numbering shall start from 0 which shall be assigned to the first byte, if any, of the data space.

The number of bytes in the data space of a file shall be referred to as the information length of the file (see 4/14.9.10).

NOTE 10 - Some record formats (see 4/14.9.7) specify records of sequences of characters delimited by a specific character sequence. If detection of these delimiter sequences requires knowledge of the character set encoding, such as would be the case if code extension characters (see 1/7.2.9.1) are used, then a Character Set Information extended attribute should be recorded.

8.9 Record structure

The information in a file may be organised as a set of records according to 4/14.9.7. The structure and attributes of these records are specified in ISO/IEC 13346-5. For the purposes of ISO/IEC 13346-5, the data space of a file is specified by 4/8.8.2 and

- if the Character Set Information extended attribute is specified for the file, then that extended attribute specifies how the bytes of the file shall be interpreted as characters,
- if the Character Set Information extended attribute is not specified for the file, then each byte of the file shall be interpreted as a single character, and a byte containing #0A shall be a LINE FEED character, a byte containing #0B shall be a VERTICAL TABULATION character, a byte containing #0C shall be a FORM FEED character, and a byte containing #0D shall be a CARRIAGE RETURN character. These interpretations shall only apply for the purposes of partitioning the file's contents into records, and need not apply to the contents of those records.

NOTE 11 - Some record formats (see ISO/IEC 13346-5) specify records delimited by a specific character sewquence.

8.10 Information Control Block (ICB)

Each recorded instance of a file shall be described by an entry in an Information Control Block (ICB). The set of entries describing the recorded instances of a file shall be described by entries in one or more ICBs. These ICBs shall form an ICB hierarchy as described in 4/8.10.1.

An ICB shall be a sequence of ICB entries recorded in an extent of logical blocks. The address or location of an ICB shall be the address of the extent. An entry of the sequence shall be one of the following:

- a direct entry, describing a recorded occurrence of a file or a set of extents
- an Indirect Entry (4/14.7), describing another ICB
- a Terminal Entry (4/14.8), indicating that there are no more entries recorded after this entry
- an unrecorded logical block, indicating that there are no more entries recorded after this entry

Each entry, other than an unrecorded logical block entry, shall specify:

- The maximum number of entries that may be recorded in the ICB in which the entry is recorded.
- The number of direct entries recorded in the ICB hierarchy prior to recording the entry.

An ICB entry shall not be recorded until all entries in that ICB with lower addresses have been recorded.

NOTE 12 - Recording an indirect entry does not imply that the ICB specified by that indirect entry is necessarily completely recorded.

The ICB may specify the strategy for building an ICB hierarchy (see 4/14.6.2). Annex 4/A specifies certain strategies; other strategies may be subject to agreement between the originator and recipient of the medium (see annex 4/A).

NOTE 13 - This Part of ISO/IEC 13346 requires a data structure that describes sequences of bytes recorded in a volume. This can be used to record a user's file, the contents of a directory or various system data. Some media, such as write-once optical disks, cannot rerecord a sector once it has been written, and thus this Part of ISO/IEC 13346 requires a data structure that can describe successive versions of regions of bytes recorded in a volume. Note that this structure is efficient on rewritable media by simply making the ICB a single direct entry. Alternatively, the same structures allow rewritable media to support a history of all versions of a file.

Whereas there is a single algorithm for traversing an ICB hierarchy, there are many algorithms or strategies for building these hierarchies.

8.10.1 ICB hierarchy

An ICB hierarchy shall be a set of ICBs descended from a root ICB. The root ICB shall be the only ICB at level 0 of an ICB hierarchy. An ICB identifying another ICB by an indirect entry shall be called a parent ICB of the identified ICB. The parent ICB shall be at level m of the ICB hierarchy and the identified ICB shall be at level $m+1$ of the ICB hierarchy.

Different ICBs may have the same parent ICB.

9 Extended attributes

An extended attribute shall specify an attribute type, an attribute subtype, and may specify attribute specific information. Extended attributes are associated with a file. All the extended attributes associated with a file shall be recorded in one or more extended attributes spaces associated with that file. The term “instances of an extended attribute” shall refer to all

extended attributes recorded in the extended attributes space of the file with identical contents of their Attribute Type and Attribute Subtype fields (see 4/14.10.2).

An attribute type shall be an integer x where $0 \leq x < 2^{32}$.
An attribute subtype shall be an integer x where $0 \leq x < 2^{8}$.
The attribute types are divided into three classes as follows:

- Attribute types 1, 3, 5, 6, 12, 2 048, and 65 536 are registered according to ISO/IEC 13800 and are recorded as specified in 4/14.10. Attribute types 2, 4, 7, 8, 9, 10 and 11 are registered according to ISO/IEC 13800 and shall not be recorded in the extended attributes space of a file. Attribute types 13 to 2 047 inclusive are reserved for reserved for registration according to ISO/IEC 13800. Attribute type 0 is reserved for future standardisation by ISO/IEC 13800.
- Attribute types 2 049 to 65 535 inclusive shall be registered according to ISO/IEC 13800, are recorded according to 4/14.10.2 and are reserved for implementation use according to ISO/IEC 13800.
- Attribute types 65 537 and above shall be registered according to ISO/IEC 13800, are recorded according to 4/14.10.2 and are reserved for application use according to ISO/IEC 13800.

There shall be

- zero or one instance of each attribute with type 1, 3, 5, 6 or 12.
- zero instances of each attribute with type 0, 2, 4, 7, 8, 9, 10 and 11.
- zero or more instances of each attribute with type 2 048 or 65 536.
- zero or more instances of each attribute with type 13 to 2047 inclusive, 2 049 to 65 535 inclusive or greater than 65 536 as specified by the registration according to ISO/IEC 13800.

The interpretation of attribute specific fields for each attribute with type

- 2 048: is specified by the `regid` (1/7.4) recorded in the Implementation Identifier field of the attribute.
- 65 536: is specified by the `regid` (1/7.4) recorded in the Application Identifier field of the attribute.
- 13 to 2 047 inclusive or 2 049 through 65 535 inclusive or greater than 65 536: is specified by the registration according to ISO/IEC 13800 of the attribute type and subtype.

If allowed by the registration of the attribute type, multiple instances of an extended attribute need not be identical; they may have different attribute specific information.

An extended attributes space of a file is one of the following:

- The Extended Attributes field of the file's File Entry.
- A file described by an ICB identified in the file's File Entry.

In each case, an extended attributes space shall be recorded according to the schema shown in figure 4/6.

```
<Extended Attribute Header Descriptor>
<Extended Attribute> 0+
```

Figure 6 - Extended attributes space schema

Extended attributes shall be recorded contiguously in three nonoverlapping areas within an extended attributes space as follows:

- The first area, starting with the first byte after the Extended Attribute Header Descriptor, is reserved for the recording of attributes defined in clauses 4/14.10.3 to 4/14.10.7.
- The second area, starting at a byte of the extended attributes space specified in the Extended Attribute Header descriptor, is reserved for the recording of attribute types 2 048 to 65 535 (see 4/14.10.8).
- The third area, starting at a byte of the extended attributes space specified in the Extended Attribute Header descriptor, is reserved for the recording of attribute types 65 536 and above (see 4/14.10.9).

The following extended attributes are defined in clause 4/14.10:

- Character Set Information

- File Times
  - File Creation Date and Time, File Deletion Date and Time, File Effective Date and Time, File Last Backup Date and Time
- Information Times
  - Information Creation Date and Time, Information Last Modification Date and Time, Information Expiration Date and Time, Information Effective Date and Time
- Alternate Permissions
- Device Specification
- Application Use
- Implementation Use

NOTE 14 - There need not be any extended attributes associated with a file. The multiple occurrences of attributes of types 2 048 and 65 536, if any, are intended to be distinguished by the contents of their Implementation Identifier and Application Identifier fields respectively. Such occurrences might have differing contents in some attribute specific fields, such as the Implementation Use or Application Use fields.

10 Partition space management

A partition has two types of space managed by this Part of ISO/IEC 13346; space ready for allocation (unallocated space), and space that may require preparation before allocation (freed space). In both cases, partition space is specified by a space set which specifies a collection of logical blocks in the partition. A space set shall be recorded as either a Space Table or as a Space Bitmap as specified in 4/10.1.

The Unallocated Space Set of a partition is a space set. If a logical block is in the Unallocated Space Set, it may be allocated for recording.

The Freed Space Set of a partition is a space set. If a logical block is in the Freed Space Set, it may be allocated for recording but may require preparation before recording as specified by the standard for recording.

The Unallocated and Freed Space Sets shall be identified by the Partition Header Descriptor (4/14.3).

10.1 Space sets

A space set shall be recorded as either a Space Table or as a Space Bitmap.

A Space Table shall be recorded as an ICB hierarchy consisting of indirect entries and unallocated space entries (see 4/14.11). The logical blocks in the space set are all the logical blocks which belong to the extents specified by the last Space Entry in the Space Table.

A Space Bitmap shall be recorded as a Space Bitmap Descriptor which includes a sequence of n bits recorded in a single extent, where n is the number of logical blocks in the partition. The value of bit s is recorded at bit $rem(s,8)$ of byte $ip(s/8)$, where byte 0 is the first byte of the extent. The space set consists of all logical blocks s such that bit s is ONE.

NOTE 15 - It is preferable that the Standard support just one type of space set. However, it is anticipated that the unallocated partition space will be updated fairly often, and that the unallocated partition space will get fragmented. Bitmaps handle the latter case efficiently but are too expensive for the former case on write-once media. In this case, we need the equivalent of an ICB which essentially records many instances of an arbitrary sequence of bytes. In fact, a Space Table is simply an ICB with direct entries specialised for recording extents of space. It is expected, but not required, that rewritable media will use space bitmaps and write-once media will use space tables.

Rewritable media may also require a second space set for logical blocks which may need to be preconditioned before recording. Some rewritable magneto-optic technology requires sectors be cleared before recording and the freed space list allows this clearing to be done asynchronously with the freeing of that space. Clearing large numbers of sectors at one time may also allow use of special hardware features such as clearing a track in one operation. For rewritable media that requires no special preprocessing for rewriting sectors, it is likely that the freed space map will be empty.

11 Partition integrity

Partition integrity specifies the status of the information recorded on the medium and shall be recorded as a Partition Integrity Table specified by the Partition Header Descriptor (see 4/14.3). This is an ICB consisting of Indirect Entries and Partition Integrity Entries (see 4/14.13) as follows:

- An Open Integrity Entry shall be recorded before any data is recorded in the partition since the last Close Integrity Entry, if any, was recorded.
- A Close Integrity Entry may be recorded only after all user data has been completely recorded and the descriptors recorded on the partition conform to this Part of ISO/IEC 13346.
- A Stable Integrity Entry may be recorded after all descriptors recorded on the partition conform to this Part of ISO/IEC 13346. However, the data in files with a 0, 5, 6, 7 or 9 in the File Type field (see 4/14.6.6) of the File Entry describing the file need not have been recorded.

NOTE 16 - The partition integrity entries provide a standard, portable and convenient way for implementations to indicate that a modified partition has had both its data and control structures properly updated. Because of optical media's large size and relatively slow access, it is particularly important to avoid unnecessary consistency checks over a partition or volume.

As an example, consider a partition mounted as a file system on a computer. When the first write request for that partition is issued, an Open Integrity Entry is recorded prior to performing the write request. When the partition is unmounted, a Close Integrity Entry is recorded after any queued write requests have been performed. Periodically, Stable Integrity Entries may be recorded after bringing up to date the data structures specified in this Part of ISO/IEC 13346. Finally, systems, such as file servers, that keep file systems mounted for long periods of time and wish to minimise the risk of a system failure and the use of lengthy recovery procedures, might adopt heuristics such as recording a Close Integrity Entry periodically or after some delay after the last write request.

12 Allocation descriptors

Allocation descriptors (see 4/14.14) specify the location, length, and type of an extent. The extent's type is one of

- recorded data,
- allocated and unrecorded space,
- space neither allocated nor recorded.

The contents of an unallocated or unrecorded extent shall be interpreted as all #00 bytes. Allocation refers to the reservation of one or more extents for current or future use, guaranteeing its availability for recording and making it unavailable for any other purpose.

A sequence of allocation descriptors shall be recorded contiguously within a field or an extent.

A field or an extent of a sequence of allocation descriptors shall be terminated by one of

- the end of the field,
- an allocation descriptor whose Extent Length field is 0,
- an allocation descriptor identifying a continuation extent in which the recording of the sequence of allocation descriptors is continued. The continuation extent shall be recorded according to the schema shown in figure 4/7.

```
[Extent of Allocation Descriptors]{
    <Allocation Extent Descriptor>
    <allocation descriptor> 1+
}
```

Figure 7 - Continuation extent schema

Allocation descriptors have an associated Information Length, which is the amount of information, in bytes, in the extent. The Extended Allocation Descriptor, or `ext_ad` (4/14.14.3), specifies the Information Length as a field; for all other allocation descriptors, the Information Length shall be the same as the length of the extent.

12.1 Description of Files

The sequence of the allocation descriptors describing the extents of a file shall be recorded as a File Body followed by a File Tail according to the schema shown in figure 4/8.

```
[File Body]{
     <allocation descriptor>(extent length is a multiple of LBS) 0+
     <allocation descriptor> 0+1
}
[File Tail]{
     <allocation descriptor>(unrecorded and allocated) 0+
}
```

Figure 8 - File extents schema

LBS denotes the logical block size. The type of allocation descriptor shall be specified by the Flags field in the ICB Tag field (see 4/14.6.8).

NOTE 17 - A sparse file, such as a large file with data recorded only at the beginning and the end of the file, might be recorded as two allocated and recorded extents separated by an unallocated and unrecorded extent.

13 Recording of descriptors

All the descriptors in this Part of ISO/IEC 13346 whose format is specified with Byte Positions (BP) shall be recorded so that the first byte of the descriptor coincides with the first byte of a logical block. All the descriptors in this Part of ISO/IEC 13346 whose format is specified with Byte Positions (BP), except for the Space Bitmap Descriptor, shall have a length no larger than the size of a logical block.

The descriptors in this Part of ISO/IEC 13346 whose format is specified with Relative Byte Positions (RBP) have no restrictions on where they may be recorded within a logical block, except that their location within a descriptor shall be specified in the description of the applicable descriptor.

When the descriptors described in this Part of ISO/IEC 13346 are recorded in a logical block, all space, if any, after the end of the last descriptor up to the end of the logical block is reserved for future standardisation and shall be recorded as all #00 bytes.

14 File Data Structures

14.1 File Set Descriptor

The File Set Descriptor shall identify a set of files and directories and shall be recorded in the format shown in figure 4/9.

BP	Length	Name	Contents
0	16	Descriptor Tag	`tag` (4/7.2)(Tag=256)
16	12	Recording Date and Time	`timestamp` (1/7.3)
28	2	Interchange Level	`Uint16` (1/7.1.3)
30	2	Maximum Interchange Level	`Uint16` (1/7.1.3)
32	4	Character Set List	`Uint32` (1/7.1.5)
36	4	Maximum Character Set List	`Uint32` (1/7.1.5)
40	4	File Set Number	`Uint32` (1/7.1.5)
44	4	File Set Descriptor Number	`Uint32` (1/7.1.5)
48	64	Logical Volume Identifier Character Set	`charspec` (1/7.2.1)
112	128	Logical Volume Identifier	`dstring` (1/7.2.12)
240	64	File Set Character Set	`charspec` (1/7.2.1)
304	32	File Set Identifier	`dstring` (1/7.2.12)
336	32	Copyright File Identifier	`dstring` (1/7.2.12)
368	32	Abstract File Identifier	`dstring` (1/7.2.12)
400	16	Root Directory ICB	`long_ad` (4/14.14.2)
416	32	Domain Identifier	`regid` (1/7.4)
448	16	Next Extent	`long_ad` (4/14.14.2)
464	48	Reserved	#00 bytes

Figure 9 - File Set Descriptor format

14.1.1 Descriptor Tag (BP 0)

The Tag Identifier field of the `tag` (4/7.2) for this descriptor shall contain 256.

14.1.2 Recording Date and Time (BP 16)

This field shall specify the date and time of the day at which this descriptor was recorded.

14.1.3 Interchange Level (BP 28)

This field shall specify the current level of medium interchange (4/15) of the file set described by this descriptor.

14.1.4 Maximum Interchange Level (BP 30)

This field shall specify the maximum value that may be specified for the Interchange Level field of this descriptor.

14.1.5 Character Set List (BP 32)

This field shall identify the character sets specified by any field, whose contents are specified to be a `charspec` (1/7.2.1), of any descriptor specified in this Part of ISO/IEC 13346 and recorded in the file set described by this descriptor.

14.1.6 Maximum Character Set List (BP 36)

The Character Set List field in this descriptor shall not specify a character set (see 1/7.2.11) not specified by the Maximum Character Set List field.

NOTE 18 - The Interchange Level, Maximum Interchange Level, Character Set List and Maximum Character Set List fields permit an implementation to:

— determine whether it can process all of the information in the file set

— restrict the recording of information in the file set so that the file set does not exceed the level given in the Maximum Interchange Level field

— restrict the recording of information in the file set so that all character sets recorded belong to the Maximum Character Set List field.

This allows a user to create a file set that can be processed when it is returned to the user.

14.1.7 File Set Number (BP 40)

This field shall specify the assigned file set number for this descriptor.

14.1.8 File Set Descriptor Number (BP 44)

This field shall specify the assigned file set descriptor number for this descriptor.

14.1.9 Logical Volume Identifier Character Set (BP 48)

This field shall specify the d-characters (1/7.2) allowed in the Logical Volume Identifier field.

If the volume is recorded according to ISO/IEC 13346-3, the contents of this field shall be identical to the contents of the Descriptor Character Set field of the Logical Volume Descriptor describing the logical volume on which the file set described by this File Set Descriptor is recorded.

14.1.10 Logical Volume Identifier (BP 112)

This field shall specify an identification of the logical volume on which the file set is recorded.

If the volume is recorded according to ISO/IEC 13346-3, the contents of this field shall be identical to the contents of the Logical Volume Identifier field of the Logical Volume Descriptor describing the logical volume on which the file set described by this File Set Descriptor is recorded.

14.1.11 File Set Character Set (BP 240)

This field shall specify the d-characters (1/7.2) allowed in certain fields of descriptors specified by this Part of ISO/IEC 13346 which have been specified as containing d-characters.

NOTE 19 - This Part of ISO/IEC 13346 does not specify the relationship between the contents of the File Set Character Set field and the Logical Volume Identifier Character Set fields or the relationship of those fields to any other fields specified by this Part of ISO/IEC 13346 or by another standard.

14.1.12 File Set Identifier (BP 304)

This field shall specify an identification of the file set described by this File Set Descriptor.

14.1.13 Copyright File Identifier (BP 336)

This field shall identify a file in the root directory containing a copyright statement for the information recorded in the file set identified by this File Set Descriptor. If the field contains all #00 bytes, then no such file is identified.

14.1.14 Abstract File Identifier (BP 368)

This field shall identify a file in the root directory containing an abstract for the information recorded in the file set identified by this File Set Descriptor. If the field contains all #00 bytes, then no such file is identified.

14.1.15 Root Directory ICB (BP 400)

This field shall specify the location of an ICB describing the root directory of the directory hierarchy associated with the file set identified by this File Set Descriptor. If the extent's length is 0, no such ICB is specified.

14.1.16 Domain Identifier (BP 416)

This field shall specify an identification of a domain which shall specify rules on the use of, and restrictions on, certain fields in descriptors subject to agreement between the originator and recipient of the medium. If this field contains all #00 bytes, then no such domain is identified. The scope of this `regid` (1/7.4) shall include all information recorded in the file set described by this descriptor.

14.1.17 Next Extent (BP 448)

This field shall specify the next extent where File Set Descriptors may be recorded. If the extent's length is 0, no such extent is specified.

14.1.18 Reserved (BP 464)

This field shall be reserved for future standardisation and all bytes shall be set to #00.

14.2 Terminating Descriptor

A Terminating Descriptor may be recorded to terminate an extent of a File Set Descriptor Sequence (see 4/8.3.1). It shall be recorded in the format shown in figure 4/10.

BP	Length	Name	Contents
0	16	Descriptor Tag	`tag` (3/7.2) (Tag=8)
16	496	Reserved	#00 bytes

Figure 10 - Terminating Descriptor format

14.2.1 Descriptor Tag (BP 0)

The Tag Identifier field of the `tag` (3/7.2) for this descriptor shall contain 8.

14.2.2 Reserved (BP 16)

This field shall be reserved for future standardisation and all bytes shall be set to #00.

14.3 Partition Header Descriptor

The Partition Header Descriptor shall specify the Unallocated Space Set, the Freed Space Set, and the Partition Integrity Table. As specified in 4/3.1, it shall be recorded with the format shown in figure 4/11.

RBP	Length	Name	Contents
0	8	Unallocated Space Table	`short_ad` (4/14.14.1)
8	8	Unallocated Space Bitmap	`short_ad` (4/14.14.1)
16	8	Partition Integrity Table	`short_ad` (4/14.14.1)
24	8	Freed Space Table	`short_ad` (4/14.14.1)
32	8	Freed Space Bitmap	`short_ad` (4/14.14.1)
40	88	Reserved	#00 bytes

Figure 11 - Partition Header Descriptor format

14.3.1 Unallocated Space Table (RBP 0)

This field shall specify the Unallocated Space Table for this partition (see 4/10). If the extent's length is 0, then no Unallocated Space Table is specified.

14.3.2 Unallocated Space Bitmap (RBP 8)

This field shall identify the extent in which the Unallocated Space Bitmap for this partition (see 4/10) is recorded. If the extent's length is 0, then no Unallocated Space Bitmap is specified.

14.3.3 Partition Integrity Table (RBP 16)

This field shall specify the Partition Integrity Table for this partition (see 4/11). If the extent's length is 0, then no Partition Integrity Table is specified.

14.3.4 Freed Space Table (RBP 24)

This field shall specify the Freed Space Table for this partition (see 4/10). If the extent's length is 0, then no Freed Space Table is specified.

14.3.5 Freed Space Bitmap (RBP 32)

This field shall identify the extent in which the Freed Space Bitmap for this partition (see 4/10) is recorded. If the extent's length is 0, then no Freed Space Bitmap is specified.

14.3.6 Reserved (RBP 40)

This field shall be reserved for future standardisation and all bytes shall be set to #00.

14.4 File Identifier Descriptor

A File Identifier Descriptor shall be recorded in the format shown in figure 4/12.

RBP	Length	Name	Contents
0	16	Descriptor Tag	`tag` (4/7.2)(Tag=257)
16	2	File Version Number	`Uint16` (1/7.1.3)
18	1	File Characteristics	`Uint8` (1/7.1.1)
19	1	Length of File Identifier (=L_FI)	`Uint8` (1/7.1.1)
20	16	ICB	`long_ad` (4/14.14.2)
36	2	Length of Implementation Use (=L_IU)	`Uint16` (1/7.1.3)
38	L_IU	Implementation Use	bytes
[L_IU+38]	L_FI	File Identifier	d-characters (1/7.2)
[L_FI+L_IU+38]	*	Padding	bytes

Figure 12 - File Identifier Descriptor format

14.4.1 Descriptor Tag (RBP 0)

The Tag Identifier field of the `tag` (4/7.2) for this descriptor shall contain 257.

14.4.2 File Version Number (RBP 16)

This field shall specify the file version number of the file specified by the File Identifier field as a number in the range 1 to 32 767 inclusive. The numbers 32 768 to 65 535 inclusive are reserved for future standardisation.

14.4.3 File Characteristics (RBP 18)

This field shall specify certain characteristics of the file as shown in figure 4/13.

Bit	Interpretation
0	Existence: If set to ZERO, shall mean that the existence of the file shall be made known to the user; If set to ONE, shall mean that the existence of the file need not be made known to the user.
1	Directory: If set to ZERO, shall mean that the file is not a directory (see 4/14.6.6); If set to ONE, shall mean that the file is a directory. NOTE 20 - If the file is a symbolic link (see 4/14.6.6), the Directory bit is set to 0.
2	Deleted: If set to ONE, shall mean this File Identifier Descriptor identifies a file that has been deleted; If set to ZERO, shall mean that this File Identifier Descriptor identifies a file that has not been deleted. NOTE 21 - The Deleted bit allows a file to be deleted from a directory by only rewriting the logical block (s) containing the File Identifier Descriptor. Note that even if the Deleted bit is set to ONE, all the descriptor's fields still need to be valid.
3	Parent: If set to ONE, shall mean that the ICB field of this descriptor identifies the ICB associated with the file in which is recorded the parent directory of the directory that this descriptor is recorded in; If set to ZERO, shall mean that the ICB field identifies the ICB associated with the file specified by this descriptor
4-7	Shall be reserved for future standardisation and all bits shall be set to ZERO.

Figure 13 - File characteristics

If the Parent bit is set to ONE, then the Directory bit shall be set to ONE.

14.4.4 Length of File Identifier (=L_FI) (RBP 19)

This field shall specify the length, in bytes, of the File Identifier field. If the Parent bit of the File Characteristics field is set to ONE, the length of the File Identifier field shall be 0.

14.4.5 ICB (RBP 20)

This field shall specify the address of an ICB describing the file. If the Delete bit of the File Characteristics field of the File Identifier Descriptor is set to ONE, the ICB field may contain all #00 bytes, in which case no ICB is specified.

14.4.6 Length of Implementation Use (=L_IU) (RBP 36)

This field shall specify the length, in bytes, of the Implementation Use field. L_IU shall be an integral multiple of 4.

14.4.7 Implementation Use (RBP 38)

If L_IU is greater than 0, this field shall specify an identification of an implementation, recorded as a `regid` (1/7.4) in the first 32 bytes of this field, which can recognise and act upon the remainder of this field, which shall be reserved for implementation use and its contents are not specified by ISO/IEC 13346. The scope of this `regid` includes the contents of this descriptor.

NOTE 22 - The scope of the `regid` does not include the file; thus file-specific information recorded in this field may become out of date when the file is modified, particularly if multiple File Identifier Descriptors refer to the file.

14.4.8 File Identifier (RBP [L_IU+38])

This field shall specify an identification for the file described by the ICB identified in the ICB field.

14.4.9 Padding (RBP [L_FI+L_IU+38])

This field shall be $4 \times ip((L\_FI+L\_IU+38+3)/4) - (L\_FI+L\_IU+38)$ bytes long and shall contain all #00 bytes.

14.5 Allocation Extent Descriptor

The Allocation Extent Descriptor shall be recorded in the format shown in figure 4/14.

BP	Length	Name	Contents
0	16	Descriptor Tag	`tag` (4/7.2)(Tag=258)
16	4	Previous Allocation Extent Location	`Uint32` (1/7.1.5)
20	4	Length of Allocation Descriptors (=L_AD)	`Uint32` (1/7.1.5)

Figure 14 - Allocation Extent Descriptor format

14.5.1 Descriptor Tag (BP 0)

The Tag Identifier field of the `tag` (4/7.2) for this descriptor shall contain 258.

14.5.2 Previous Allocation Extent Location (BP 16)

This field shall specify the address, within the partition the descriptor is recorded on, of the previous allocation extent. If the extent's length is 0, the previous allocation extent is not specified.

14.5.3 Length of Allocation Descriptors (=L_AD) (BP 20)

This field specifies the length, in bytes, of the allocation descriptors recorded after this descriptor.

14.6 ICB Tag

All the entries in an ICB shall have a common format; a `tag` (4/7.2), followed by an `icbtag` as shown in figure 4/15, followed by a part that is unique to each type of entry and is described in the definition of that entry.

RBP	Length	Name	Contents
0	4	Prior Recorded Number of Direct Entries	`Uint32` (1/7.1.5)
4	2	Strategy Type	`Uint16` (1/7.1.3)
6	2	Strategy Parameter	bytes
8	2	Maximum Number of Entries	`Uint16` (1/7.1.3)
10	1	Reserved	#00 byte
11	1	File Type	`Uint8` (1/7.1.1)
12	6	Parent ICB Location	`lb_addr` (4/7.1)
18	2	Flags	`Uint16` (1/7.1.3)

Figure 15 - `icbtag` format

14.6.1 Prior Recorded Number of Direct Entries (RBP 0)

This field specifies the number of Direct Entries recorded in this ICB hierarchy prior to this entry.

14.6.2 Strategy Type (RBP 4)

This field shall specify the strategy for building the ICB hierarchy of which the ICB is a member. The strategies are specified by a number as shown in figure 4/16.

Type	Interpretation
0	The strategy is not specified by this clause.
1	The strategy is specified in 4/A.2.
2	The strategy is specified in 4/A.3.
3	The strategy is specified in 4/A.4.
4	The strategy is specified in 4/A.5.
5-4 095	Reserved for future standardisation.
4 096-65 535	The interpretation of the strategy shall be subject to agreement between the originator and recipient of the medium.

Figure 16 - ICB strategies

14.6.3 Strategy Parameter (RBP 6)

This field shall be interpreted according to the strategy specified by the Strategy Type field.

14.6.4 Maximum Number of Entries (RBP 8)

This field specifies the maximum number of entries, including both direct and indirect, that may be recorded in this ICB. This field shall be greater than 0.

14.6.5 Reserved (RBP 10)

This field shall be reserved for future standardisation and shall be set to 0.

14.6.6 File Type (RBP 11)

This field shall specify the type of the file as shown in figure 4/17.

Type	Interpretation
0	Shall mean that the interpretation of the file is not specified by this field
1	Shall mean that this is an Unallocated Space Entry (see 4/14.11)
2	Shall mean that this is a Partition Integrity Entry (see 4/14.13)
3	Shall mean that this is an Indirect Entry (see 4/14.7)
4	Shall mean that the file is a directory (see 4/8.6)
5	Shall mean that the file shall be interpreted as a sequence of bytes, each of which may be randomly accessed
6	Shall mean that the file is a block special device file as specified by ISO/IEC 9945-1
7	Shall mean that the file is a character special device file as specified by ISO/IEC 9945-1
8	Shall mean that the file is for recording Extended Attributes as described in 4/9
9	Shall mean that the file is a FIFO file as specified by ISO/IEC 9945-1
10	Shall mean that the file shall be interpreted according to the `C_ISSOCK` file type identified by ISO/IEC 9945-1
11	Shall mean that this is a Terminal Entry (see 4/14.8)
12	Shall mean that the file is a symbolic link and that its content is a pathname (see 4/8.7) for a file or directory
13-255	Reserved for future standardisation

Figure 17 - File Types

The interpretation of the content of files of File Types 0 and 5 shall be subject to agreement between the originator and recipient of the medium.

14.6.7 Parent ICB Location (RBP 12)

This field shall specify the location of the ICB which contains an indirect entry specifying the ICB that this descriptor is recorded in. If this field contains 0, no such ICB is specified.

14.6.8 Flags (RBP 18)

This field shall specify recording information about the file as shown in figure 4/18.

Bit	Interpretation
0-2	Shall be interpreted as a 3-bit unsigned binary number as follows. The value 0 shall mean that Short Allocation Descriptors (4/14.14.1) are used. The value 1 shall mean that Long Allocation Descriptors (4/14.14.2) are used. The value 2 shall mean that Extended Allocation Descriptors (4/14.14.3) are used. The value 3 shall mean that the file shall be treated as though it had exactly one allocation descriptor describing an extent which starts with the first byte of the Allocation Descriptors field and has a length, in bytes, recorded in the Length of Allocation Descriptors field. The values of 4-7 are reserved for future standardisation.
3	If the file is not a directory, this bit shall be reserved for future standardisation and set to ZERO. If the file is a directory and this bit is set to ONE, the directory shall be sorted according to 4/8.6.1. If the file is a directory and this bit is set to ZERO, the directory need not be sorted according to 4/8.6.1.
4	Non-relocatable: If set to ZERO, shall mean that there are no restrictions on how the allocation descriptors specifying the file's data may be modified; If set to ONE, the allocation descriptors shall not be modified such that either the address of an extent of the file is changed or that the recorded length of an extent is reduced.
5	Archive: This bit shall be set to ONE when the file is created or is written. This bit shall be set to ZERO in an implementation-dependent manner.
6	Setuid: This bit shall be interpreted as the `S_ISUID` bit as specified in ISO/IEC 9945-1.
7	Setgid: This bit shall be interpreted as the `S_ISGID` bit as specified in ISO/IEC 9945-1.
8	Sticky: This bit shall be interpreted as the `S_ISVTX` bit as specified in ISO/IEC 9945-1.
9	Contiguous: If set to ZERO, then an extent of a file need not begin at the first logical block after the last logical block of the preceding extent of the file; If set to ONE, then each extent of a file shall begin at the first logical block after the last logical block of the preceding extent of the file.
10	System: This bit shall be reserved for implementation use.
11	Transformed: If set to ZERO, shall mean that the recorded bytes of the data space of the file are those supplied by the user. If set to ONE, shall mean that the bytes supplied by the user have been transformed in a manner not specified by ISO/IEC 13346 prior to recording.
12	Multi-versions: If the file is not a directory, this bit shall be reserved for future standardisation and shall be set to ZERO. If the file is a directory and the bit is set to ZERO, then no two File Identifier Descriptors recorded in the directory shall have the same contents of their File Identifier fields. If the file is a directory and the bit is set to ONE, then there may be two or more File Identifier Descriptors recorded in the directory with identical contents of their File Identifier fields.
13-15	Shall be reserved for future standardisation and all bits shall be set to ZERO.

Figure 18 - File characteristics

14.7 Indirect Entry

An Indirect Entry shall be recorded in the format shown in figure 4/19.

BP	Length	Name	Contents
0	16	Descriptor Tag	`tag` (4/7.2) (Tag=259)
16	20	ICB Tag	`icbtag` (4/14.6) (Type=3)
36	16	Indirect ICB	`long_ad` (4/14.14.2)

Figure 19 - Indirect Entry format

14.7.1 Descriptor Tag (BP 0)

The Tag Identifier field of the `tag` (4/7.2) for this descriptor shall contain 259.

14.7.2 ICB Tag (BP 16)

The File Type field of the `icbtag` (4/14.6) for this descriptor shall contain 3.

14.7.3 Indirect ICB (BP 36)

This field shall specify the address of another ICB.

14.8 Terminal Entry

A Terminal Entry shall be recorded in the format shown in figure 4/20.

BP	Length	Name	Contents
0	16	Descriptor Tag	`tag` (4/7.2) (Tag=260)
16	20	ICB Tag	`icbtag` (4/14.6) (Type=11)

Figure 20 - Terminal Entry format

14.8.1 Descriptor Tag (BP 0)

The Tag Identifier field of the `tag` (4/7.2) for this descriptor shall contain 260.

14.8.2 ICB Tag (BP 16)

The File Type field of the `icbtag` (4/14.6) for this descriptor shall contain 11.

14.9 File Entry

The File Entry is a direct entry recorded in an ICB in the format shown in figure 4/21 for File Types 0 and 4-10 as specified in 4/14.6.6.

BP	Length	Name	Contents
0	16	Descriptor Tag	`tag` (4/7.2)(Tag=261)
16	20	ICB Tag	`icbtag` (4/14.6)
36	4	Uid	`Uint32` (1/7.1.5)
40	4	Gid	`Uint32` (1/7.1.5)
44	4	Permissions	`Uint32` (1/7.1.5)
48	2	File Link Count	`Uint16` (1/7.1.3)
50	1	Record Format	`Uint8` (1/7.1.1)
51	1	Record Display Attributes	`Uint8` (1/7.1.1)
52	4	Record Length	`Uint32` (1/7.1.5)
56	8	Information Length	`Uint64` (1/7.1.7)
64	8	Logical Blocks Recorded	`Uint64` (1/7.1.7)
72	12	Access Date and Time	`timestamp` (1/7.3)
84	12	Modification Date and Time	`timestamp` (1/7.3)
96	12	Attribute Date and Time	`timestamp` (1/7.3)
108	4	Checkpoint	`Uint32` (1/7.1.5)
112	16	Extended Attribute ICB	`long_ad` (4/14.14.2)
128	32	Implementation Identifier	`regid` (1/7.4)
160	8	Unique Id	`Uint64` (1/7.1.7)
168	4	Length of Extended Attributes (=L_EA)	`Uint32` (1/7.1.5)
172	4	Length of Allocation Descriptors (=L_AD)	`Uint32` (1/7.1.5)
176	L_EA	Extended Attribute s	bytes
[L_EA+176]	L_AD	Allocation descriptors	bytes

Figure 21 - File Entry format

14.9.1 Descriptor Tag (BP 0)

The Tag Identifier field of the `tag` (4/7.2) for this descriptor shall contain 261.

14.9.2 ICB Tag (BP 16)

The File Type field of the `icbtag` (4/14.6) for this descriptor shall be recorded as specified in 4/14.6.6.

14.9.3 Uid (BP 36)

This field shall specify the user ID of the owner of the file.

NOTE 23 - Originating systems that do not support the notion of user IDs will probably use an arbitrary user ID (and group ID). For various historical reasons, it is recommended such systems do not choose 0 for these IDs.

14.9.4 Gid (BP 40)

This field shall specify the group ID of the owner of the file.

14.9.5 Permissions (BP 44)

This field shall specify the access allowed to the current file for certain classes of users as follows:

- If the user's user ID is the same as the Uid field, then bits 10-14 shall apply.
- Otherwise, if the user's group ID is the same as the Gid field, then bits 5-9 shall apply.
- Otherwise, bits 0-4 shall apply.

The allowed access is shown in figure 4/22.

Bit	Interpretation
0	Other: If set to ZERO, shall mean that the user may not execute the file; If set to ONE, shall mean that the user may execute the file.
1	Other: If set to ZERO, shall mean that the user may not write the file; If set to ONE, shall mean that the user may write the file.
2	Other: If set to ZERO, shall mean that the user may not read the file; If set to ONE, shall mean that the user may read the file.
3	Other: If set to ZERO, shall mean that the user may not change any attributes of the file; If set to ONE, shall mean that the user may change attributes of the file.
4	Other: If set to ZERO, shall mean that the user may not delete the file; If set to ONE, shall mean that the user may delete the file.
5	Group: If set to ZERO, shall mean that the user may not execute the file; If set to ONE, shall mean that the user may execute the file.
6	Group: If set to ZERO, shall mean that the user may not write the file; If set to ONE, shall mean that the user may write the file.
7	Group: If set to ZERO, shall mean that the user may not read the file; If set to ONE, shall mean that the user may read the file.
8	Group: If set to ZERO, shall mean that the user may not change any attributes of the file; If set to ONE, shall mean that the user may change attributes of the file.
9	Group: If set to ZERO, shall mean that the user may not delete the file; If set to ONE, shall mean that the user may delete the file.
10	Owner: If set to ZERO, shall mean that the user may not execute the file; If set to ONE, shall mean that the user may execute the file.
11	Owner: If set to ZERO, shall mean that the user may not write the file; If set to ONE, shall mean that the user may write the file.
12	Owner: If set to ZERO, shall mean that the user may not read the file; If set to ONE, shall mean that the user may read the file.
13	Owner: If set to ZERO, shall mean that the user may not change any attributes of the file; If set to ONE, shall mean that the user may change attributes of the file.
14	Owner: If set to ZERO, shall mean that the user may not delete the file; If set to ONE, shall mean that the user may delete the file.
15-31	Reserved: Shall be set to ZERO.

Figure 22 - Allowed access

NOTE 24 - File access schemes are subject to agreement between the originator and recipient of the medium as the meanings of both user IDs and group IDs are implementation dependent; indeed, the permission and file access models of the receiving and originating systems may be incompatible.

The question of how to interpret permissions on systems which do not support user IDs and group IDs is outside the scope of this Part of ISO/IEC 13346. However, if a system uses the Uid, Gid and Permissions fields, it is recommended that such systems use and set all three (owner, group, other) sets of permissions. It is also recommended that the Uid, Gid and Permissions fields be mapped to the appropriate fields in the implementation.

14.9.6 File Link Count (BP 48)

This field shall specify the number of File Identifier Descriptors identifying this ICB.

NOTE 25 - Implementations should not blindly copy the contents of this field from the source File Entry when copying a directory hierarchy onto a volume. This field should only be incremented as links are made.

14.9.7 Record Format (BP 50)

This field shall specify a number identifying the format of the information in the file as shown in figure 4/23.

Number	Interpretation
0	Shall mean that the structure of the information recorded in the file is not specified by this field.
1	Shall mean that the information in the file is a sequence of padded fixed-length records (see 5/9.2.1).
2	Shall mean that the information in the file is a sequence of fixed-length records (see 5/9.2.2).
3	Shall mean that the information in the file is a sequence of variable-length-8 records (see 5/9.2.3.1).
4	Shall mean that the information in the file is a sequence of variable-length-16 records (see 5/9.2.3.2).
5	Shall mean that the information in the file is a sequence of variable-length-16-MSB records (see 5/9.2.3.3).
6	Shall mean that the information in the file is a sequence of variable-length-32 records (see 5/9.2.3.4).
7	Shall mean that the information in the file is a sequence of stream-print records (see 5/9.2.4).
8	Shall mean that the information in the file is a sequence of stream-LF records (see 5/9.2.5).
9	Shall mean that the information in the file is a sequence of stream-CR records (see 5/9.2.6).
10	Shall mean that the information in the file is a sequence of stream-CRLF records (see 5/9.2.7).
11	Shall mean that the information in the file is a sequence of stream-LFCR records (see 5/9.2.8).
12-255	Reserved for future standardisation.

Figure 23 - Information format

If the File Type field of the ICB Tag field contains 1, 2, 3, 4, 8, 11 or 12, the Record Format field shall contain 0.

14.9.8 Record Display Attributes (BP 51)

This field shall specify certain intended display attributes of the records in a file as shown in figure 4/24.

Attributes	Interpretation
0	Shall mean that the manner of display of a record is not specified by this field.
1	Shall mean that each record shall be displayed on a character-imaging device according to 5/9.3.1.
2	Shall mean that each record shall be displayed on a character-imaging device according to 5/9.3.2.
3	Shall mean that each record shall be displayed on a character-imaging device according to 5/9.3.3.
4-255	Reserved for future standardisation.

Figure 24 - Record display characteristics

14.9.9 Record Length (BP 52)

If the Record Format field contains the number 0, the interpretation of the Record Length field is subject to agreement between the originator and recipient of the medium.

If the Record Format field contains either 1 or 2, the Record Length field shall specify the length, in bytes, of each record in the file.

If the Record Format field contains a number in the range 3-11 inclusive, the Record Length field shall specify the maximum length, in bytes, of a record that may be recorded in the file.

14.9.10 Information Length (BP 56)

The file size in bytes. This shall be equal to the sum of the Information Lengths of the allocation descriptors for the body of the file (see 4/8.8.2 and 4/12).

NOTE 26 - This is not necessarily the number of recorded bytes. There may be unrecorded extents or there may be `ext_ad` (4/14.14.3) allocation descriptors.

14.9.11 Logical Blocks Recorded (BP 64)

The number of recorded logical blocks specified by the allocation descriptors for the body of the file (see 4/12.1).

14.9.12 Access Date and Time (BP 72)

This field shall specify the most recent date and time of the day of file creation or read access to the file prior to recording this File Entry. This date and time shall not be earlier than the File Creation Date and Time specified in the File Times Extended Attribute, if any.

NOTE 27 - This departs a little from the interpretation in ISO/IEC 9945-1 in that read accesses since this File Entry was recorded are ignored. This is intended to reduce updates on write-once media.

14.9.13 Modification Date and Time (BP 84)

This field shall specify the most recent date and time of the day of file creation or write access to the file. This date and time shall not be earlier than the File Creation Date and Time specified in the File Times Extended Attribute, if any.

14.9.14 Attribute Date and Time (BP 96)

This field shall specify the most recent date and time of the day of file creation or modification of the attributes of the file. This date and time shall not be earlier than the File Creation Date and Time specified in the File Times Extended Attribute, if any.

14.9.15 Checkpoint (BP 108)

This field shall contain 1 for the first instance of a file and shall be incremented by 1 when directed to do so by the user. This Part of ISO/IEC 13346 does not specify any relationship between the Checkpoint field and the File Version Number field of the directory descriptor identifying the file.

NOTE 28 - This field allows the user to label sequences of instances of a file with a monotonic increasing numeric tag. It has an interpretation similar to that of the File Version Number but is not part of the file's identification and need not have the same value as the File Version Number. The motivation is that the user will often have no control over when an implementation will flush a file to disk (and thus creating a new instance). In this situation, the user may simply increment the Checkpoint field when it is appropriate.

14.9.16 Extended Attribute ICB (BP 112)

This field shall specify the ICB describing the extended attribute file for the file. If the extent's length is 0, no such ICB is specified.

14.9.17 Implementation Identifier (BP 128)

This field shall specify an identification of an implementation which can recognise and act upon the contents of the Implementation Use field, if any, of the allocation descriptors for this File Entry. If this field contains all #00 bytes, then no such implementation is identified. The scope of this `regid` includes the contents of the descriptors that specify the contents and attributes of the file described by this descriptor.

14.9.18 Unique Id (BP 160)

This field shall specify a numeric identifier for this file. All File Entries with the same contents of this field shall describe the same file or directory.

14.9.19 Length of Extended Attributes (=L_EA) (BP 168)

This field shall specify the length, in bytes, of the Extended Attributes field. L_EA shall be an integral multiple of 4.

14.9.20 Length of Allocation Descriptors (=L_AD) (BP 172)

This field shall specify the length, in bytes, of the Allocation Descriptors field.

14.9.21 Extended Attributes (BP 176)

This field shall contain an extended attributes space (see 4/9). The recorded extended attributes shall occupy at most L_EA bytes and any unused bytes shall be set to #00.

14.9.22 Allocation Descriptors (BP [L_EA+176])

This field shall be a sequence of allocation descriptors recorded as specified in 4/12.1. Any such allocation descriptor which is specified as unrecorded and unallocated (see 4/14.14.1.1) shall have its Extent Location field set to 0.

14.10 Extended Attributes

In this clause, the term "current file " shall refer to the file that the extended attribute is associated with.

14.10.1 Extended Attribute Header Descriptor

The Extended Attribute Header Descriptor shall be recorded in the format shown in figure 4/25.

RBP	Length	Name	Contents
0	16	Descriptor Tag	`tag` (4/7.2)(Tag=262)
16	4	Implementation Attributes Location	`Uint32` (1/7.1.5)
20	4	Application Attributes Location	`Uint32` (1/7.1.5)

Figure 25 - Extended Attribute Header Descriptor format

14.10.1.1 Descriptor Tag (RBP 0)

The Tag Identifier field of the `tag` (4/7.2) for this descriptor shall contain 262.

14.10.1.2 Implementation Attributes Location (RBP 16)

This field shall specify the start of the implementation use extended attributes as a byte offset from the start of an extended attributes space in which extended attributes of the current file shall be recorded.

14.10.1.3 Application Attributes Location (RBP 20)

This field shall specify the start of the application use extended attributes as a byte offset from the start of an extended attributes space in which extended attributes of the current file shall be recorded.

14.10.2 Generic format

An Extended Attribute shall be recorded in the format shown in figure 4/26. The specification for each extended attribute shall specify the interpretation of the Attribute Subtype and Attribute Data fields of the extended attribute.

RBP	Length	Name	Contents
0	4	Attribute Type	`Uint32` (1/7.1.5)
4	1	Attribute Subtype	`Uint8` (1/7.1.1)
5	3	Reserved	#00 bytes
8	4	Attribute Length (=A_L)	`Uint32` (1/7.1.5)
12	A_L-12	Attribute Data	bytes

Figure 26 - Generic extended attribute format

14.10.2.1 Attribute Type (RBP 0)

This field shall specify the type of the extended attribute.

14.10.2.2 Attribute Subtype (RBP 4)

This field shall specify the subtype of the extended attribute.

14.10.2.3 Reserved (RBP 5)

This field shall be reserved for future standardisation and all bytes shall be set to #00.

14.10.2.4 Attribute Length (=A_L) (RBP 8)

This field shall specify the length of the entire extended attribute.

NOTE 29 - It is recommended that the extended attribute length be an integral multiple of 4.

14.10.2.5 Attribute Data (RBP 12)

The interpretation of this field shall depend on the value of the Attribute Type field.

NOTE 30 - The only meaning for the Attribute Length field (A_L) is the distance in bytes from the start of the extended attribute to the start of the next, if any, extended attribute. The only deduction one can make is that the amount of attribute specific data is not greater than A_L-12. It is recommended that extended attributes with variable-sized data record the data length immediately after the Attribute Length field.

This scheme allows for arbitrary alignment of the attributes and their data. In particular, there may be padding bytes between the end of the data for an attribute and the start of the next attribute. Implementations are not required to preserve any attribute alignments.

14.10.3 Character Set Information

The Character Set Information Extended Attribute shall be recorded in the format shown in figure 4/27. The Character Set Information Extended Attribute may be used to specify the coded character sets used in interpreting the contents of the current file.

RBP	Length	Name	Contents
0	4	Attribute Type	`Uint32` (1/7.1.5) = 1
4	1	Attribute Subtype	`Uint8` (1/7.1.1) = 1
5	3	Reserved	#00 bytes
8	4	Attribute Length	`Uint32` (1/7.1.5)
12	4	Escape Sequences Length (=ES_L)	`Uint32` (1/7.1.5)
16	1	Character Set Type	`Uint8` (1/7.1.1)
17	ES_L	Escape Sequences	bytes

Figure 27 - Character Set Information Extended Attribute format

14.10.3.1 Attribute Type (RBP 0)

This field shall specify 1.

14.10.3.2 Attribute Subtype (RBP 4)

This field shall specify 1. All other values are reserved for future standardisation.

14.10.3.3 Reserved (RBP 5)

This field shall be reserved for future standardisation and all bytes shall be set to #00.

14.10.3.4 Attribute Length (RBP 8)

This field shall specify the length of the entire extended attribute.

NOTE 31 - It is recommended that the extended attribute length be an integral multiple of 4.

14.10.3.5 Escape Sequences Length (=ES_L) (RBP 12)

This field shall specify the length in bytes of the Escape Sequences field.

14.10.3.6 Character Set Type (RBP 16)

This field shall specify the character set type as specified in 1/7.2.1, except that any information that would be recorded in the Character Set Information field shall instead be recorded in the Escape Sequences field.

14.10.3.7 Escape Sequences (RBP 17)

This field shall specify one or more escape sequences, control sequences or both escape sequences and control sequences according to ISO/IEC 2022 and ISO/IEC 6429 that designate and implicitly invoke the coded character sets to be used in interpreting the conents of the current file in an 8-bit environment according to ISO/IEC 2022 or ISO/IEC 10646-1. These sequences shall be recorded contiguously from the start of the field and any unused bytes shall be set to #00.

14.10.4 Alternate Permissions

The Alternate Permissions extended attribute specifies fields that can be used to support the file access permission scheme of ISO 9660 for the current file. It shall be recorded in the format shown in figure 4/28.

NOTE 32 - This extended attribute is an extension of the permissions field in ISO 9660 to allow for the case of writing information. In addition, it eliminates the inconsistencies of the specification in ISO 9660.

If the user's user ID is the same as the Owner Identification field and the user's group ID is the same as the Group Identification field, the user shall be treated as the owner of the file.

RBP	Length	Name	Contents
0	4	Attribute Type	Uint32 (1/7.1.5) = 3
4	1	Attribute Subtype	Uint8 (1/7.1.1) = 1
5	3	Reserved	#00 bytes
8	4	Attribute Length	Uint32 (1/7.1.5)
12	2	Owner Identification	Uint16 (1/7.1.3)
14	2	Group Identification	Uint16 (1/7.1.3)
16	2	Permission	Uint16 (1/7.1.3)

Figure 28 - Alternate Permissions Extended Attribute format

14.10.4.1 Attribute Type (RBP 0)

This field shall specify 3.

14.10.4.2 Attribute Subtype (RBP 4)

This field shall specify 1. All other values are reserved for future standardisation.

14.10.4.3 Reserved (RBP 5)

This field shall be reserved for future standardisation and all bytes shall be set to #00.

14.10.4.4 Attribute Length (RBP 8)

This field shall specify the length of the entire extended attribute.

NOTE 33 - It is recommended that the extended attribute length be an integral multiple of 4.

14.10.4.5 Owner Identification (RBP 12)

This field shall specify as a 16-bit number an identification of the owner of the file who is a member of the group identified by the Group Identification field of this extended attribute.

If the number in this field is 0, this shall indicate that there is no owner identification specified for the file. In this case, the Group Identification field shall be set to 0.

14.10.4.6 Group Identification (RBP 14)

This field shall specify as a 16-bit number an identification of the group of which the owner of the file is a member.

For this number, values from 1 to a value subject to agreement between the originator and recipient of the medium shall identify the group as belonging to the class of user referred to as System.

If the number in this field is 0, this shall indicate that there is no group identification specified for the file. In this case, the Owner Identification field shall be set to 0.

14.10.4.7 Permissions (RBP 16)

This field shall specify, for certain classes of users, if read, write, execute, and delete access is allowed for the file. The desired access shall be given if at least one of the following conditions is true:

- the user's user ID is the same as the Owner Identification field and the user's group ID is the same as the Group Identification field and bits 4-7 allow the desired access,
- bits 12-15 allow the desired access,
- the user's group ID is the same as the Group Identification field and bits 8-11 allow the desired access,
- if the user's group ID identifies a group of the System class of user and bits 0-3 allow the desired access.

The allowed access is shown in figure 4/29.

Bit	Interpretation
0	If set to ZERO, shall mean that a user who is a member of a group of the System class of user may read the file. If set to ONE, shall mean that read access is not allowed by this bit.
1	If set to ZERO, shall mean that a user who is a member of a group of the System class of user may write the file. If set to ONE, shall mean that write access is not allowed by this bit.
2	If set to ZERO, shall mean that a user who is a member of a group of the System class of user may execute the file. If set to ONE, shall mean that execute access is not allowed by this bit.
3	If set to ZERO, shall mean that a user who is a member of a group of the System class of user may delete the file. If set to ONE, shall mean that delete access is not allowed by this bit.
4	If set to ZERO, shall mean that the owner may read the file. If set to ONE, shall mean that read access is not allowed by this bit.
5	If set to ZERO, shall mean that the owner may write the file. If set to ONE, shall mean that write access is not allowed by this bit.
6	If set to ZERO, shall mean that the owner may execute the file. If set to ONE, shall mean that execute access is not allowed by this bit.
7	If set to ZERO, shall mean that the owner may delete the file. If set to ONE, shall mean that delete access is not allowed by this bit.
8	If set to ZERO, shall mean that any user who has a group ID that is the same as the Group Identification field may read the file. If set to ONE, shall mean that read access is not allowed by this bit.
9	If set to ZERO, shall mean that any user who has a group ID that is the same as the Group Identification field may write the file. If set to ONE, shall mean that write access is not allowed by this bit.
10	If set to ZERO, shall mean that any user who has a group ID that is the same as the Group Identification field may execute the file. If set to ONE, shall mean that execute access is not allowed by this bit.
11	If set to ZERO, shall mean that any user who has a group ID that is the same as the Group Identification field may delete the file. If set to ONE, shall mean that delete access is not allowed by this bit.
12	If set to ZERO, shall mean that any user may read the file. If set to ONE, shall mean that read access is not allowed by this bit.
13	If set to ZERO, shall mean that any user may write the file. If set to ONE, shall mean that write access is not allowed by this bit.
14	If set to ZERO, shall mean that any user may execute the file. If set to ONE, shall mean that execute access is not allowed by this bit.
15	If set to ZERO, shall mean that any user may delete the file. If set to ONE, shall mean that delete access is not allowed by this bit.

Figure 29 - Allowed access

NOTE 34 - File access schemes are subject to agreement between the originator and recipient of the medium as the meanings of both user IDs and group IDs are implementation dependent; indeed, the permission and file access models of the receiving and originating systems may be incompatible.

The question of how to interpret permissions on systems which do not support user IDs and group IDs is outside the scope of ISO/IEC 13346-4. However, if a system uses the Alternate Permissions extended attribute, it is recommended that such systems use and set all four (system, owner, group, other) sets of permissions. It is also recommended that the Owner Identification, Group Identification and Permission fields be mapped to the appropriate fields in the implementation.

14.10.5 File Times Extended Attribute

The File Times Extended Attribute specifies certain dates and times for the current file and shall be recorded as shown in figure 4/30.

RBP	Length	Name	Contents
0	4	Attribute Type	`Uint32` (1/7.1.5) = 5
4	1	Attribute Subtype	`Uint8` (1/7.1.1) = 1
5	3	Reserved	#00 bytes
8	4	Attribute Length	`Uint32` (1/7.1.5
12	4	Data Length(=D_L)	`Uint32` (1/7.1.5)
16	4	File Time Existence	`Uint32` (1/7.1.5)
20	D_L	File Times	bytes

Figure 30 - File Times Extended Attribute format

14.10.5.1 Attribute Type (RBP 0)

This field shall specify 5.

14.10.5.2 Attribute Subtype (RBP 4)

This field shall specify 1. All other values are reserved for future standardisation.

14.10.5.3 Reserved (RBP 5)

This field shall be reserved for future standardisation and all bytes shall be set to #00.

14.10.5.4 Attribute Length (RBP 8)

This field shall specify the length of the entire extended attribute.

NOTE 35 - It is recommended that the extended attribute length be an integral multiple of 4.

14.10.5.5 Data Length(=D_L) (RBP 12)

This field shall contain the number of bytes used to record the dates and times specified by the File Time Existence field.

14.10.5.6 File Time Existence (RBP 16)

This field shall specify which dates and times shall be recorded in the File Times field. A bit in this field corresponds to a particular date and time as shown in figure 4/31. If the bit is ZERO, then that date and time shall not be recorded. If the bit is ONE, then that date and time shall be recorded. Bits not specified in figure 4/31 are reserved for future standardisation and shall be set to ZERO.

Bit	Interpretation
0	File Creation Date and Time: the date and time of the day at which the file was created.
2	File Deletion Date and Time: the date and time of the day after which the file may be deleted. If the bit is ZERO, the file may not be deleted unless requested by the user.
3	File Effective Date and Time: the date and time of the day after which the file may be used. If the bit is ZERO, the file may be used at once.
5	File Last Backup Date and Time: the date and time of the day at which the file was last backed up.

Figure 31 - File Times

NOTE 36 - Bits 1 and 4 are deliberately unused for compatibility with ISO/IEC 13490-2. Attribute type 5 in ISO/IEC 13490-2 also specifies File Last Access Date and Time and File Modification Date and Time.. Those dates and times are specified in the File Entry (see 4/14.9.12 and 4/ 14.9.13) of ISO/IEC 13346-4.

14.10.5.7 File Times (RBP 20)

The dates and times specified in the File Times Existence field shall be recorded contiguously in this field, each as a `timestamp` (1/7.3), in ascending order of their bit positions.

14.10.6 Information Times Extended Attribute

The Information Times Extended Attribute specifies certain dates and times for the information in the current file and shall be recorded as shown in figure 4/32.

RBP	Length	Name	Contents
0	4	Attribute Type	`Uint32` (1/7.1.5) = 6
4	1	Attribute Subtype	`Uint8` (1/7.1.1) = 1
5	3	Reserved	#00 bytes
8	4	Attribute Length	`Uint32` (1/7.1.5)
12	4	Data Length(=D_L)	`Uint32` (1/7.1.5)
16	4	Information Time Existence	`Uint32` (1/7.1.5)
20	D_L	Information Times	bytes

Figure 32 - Information Times Extended Attribute format

14.10.6.1 Attribute Type (RBP 0)

This field shall specify 6.

14.10.6.2 Attribute Subtype (RBP 4)

This field shall specify 1. All other values are reserved for future standardisation.

14.10.6.3 Reserved (RBP 5)

This field shall be reserved for future standardisation and all bytes shall be set to #00.

14.10.6.4 Attribute Length (RBP 8)

This field shall specify the length of the entire extended attribute.

NOTE 37 - It is recommended that the extended attribute length be an integral multiple of 4.

14.10.6.5 Data Length(=D_L) (RBP 12)

This field shall contain the number of bytes used to record the dates and times specified by the Information Time Existence field.

14.10.6.6 Information Time Existence (RBP 16)

This field shall specify which dates and times shall be recorded in the Information Times field. A bit in this field corresponds to a particular date and time as shown in figure 4/33. If the bit is ZERO, then that date and time shall not be recorded. If the bit is ONE, then that date and time shall be recorded. Bits not specified in figure 4/33 are reserved for future standardisation and shall be set to ZERO.

Bit	Interpretation
0	Information Creation Date and Time: the date and time of the day at which the information in the file was created.
1	Information Last Modification Date and Time: the date and time of the day at which the information in the file was last modified.
2	Information Expiration Date and Time: the date and time of the day after which the information in the file may be regarded as obsolete. If the bit is ZERO, the information in the file shall not be regarded as obsolete unless requested by the user.
3	Information Effective Date and Time: the date and time of the day after which the information in the file may be used. If the bit is ZERO, the information in the file may be used at once.

Figure 33 - Information Times

14.10.6.7 Information Times (RBP 20)

The dates and times specified in the Information Times Existence field shall be recorded contiguously in this field, each as a `timestamp` (1/7.3), in ascending order of their bit positions.

14.10.7 Device Specification

The Device Specification Extended Attribute shall be recorded in the format shown in figure 4/34. It shall specify a device subject to agreement between the originator and recipient of the medium.

RBP	Length	Name	Contents
0	4	Attribute Type	Uint32 (1/7.1.5) = 12
4	1	Attribute Subtype	Uint8 (1/7.1.1) = 1
5	3	Reserved	#00 bytes
8	4	Attribute Length	Uint32 1/7.1.5)
12	4	Implementation Use Length (=IU_L)	Uint32 (1/7.1.5)
16	4	Major Device Identification	Uint32 (1/7.1.5)
20	4	Minor Device Identification	Uint32 (1/7.1.5)
24	IU_L	Implementation Use	bytes

Figure 34 - Device Specification Extended Attribute format

14.10.7.1 Attribute Type (RBP 0)

This field shall specify 12.

14.10.7.2 Attribute Subtype (RBP 4)

This field shall specify 1. All other values are reserved for future standardisation.

14.10.7.3 Reserved (RBP 5)

This field shall be reserved for future standardisation and all bytes shall be set to #00.

14.10.7.4 Attribute Length (RBP 8)

This field shall specify the length of the entire extended attribute.

NOTE 38 - It is recommended that the extended attribute length be an integral multiple of 4.

14.10.7.5 Implementation Use Length (=IU_L) (RBP 12)

This field shall specify the length in bytes of the Implementation Use field.

14.10.7.6 Major Device Identification (RBP 16)

This field may be used to specify a device. The contents of this field shall be subject to agreement between the originator and recipient of the medium.

14.10.7.7 Minor Device Identification (RBP 20)

This field may be used to specify a device. The contents of this field shall be subject to agreement between the originator and recipient of the medium.

14.10.7.8 Implementation Use (RBP 24)

If IU_L is greater than 0, this field shall specify an identification of an implementation, recorded as a `regid` (1/7.4) in the first 32 bytes of this field, which can recognise and act upon the remainder of this field, which shall be reserved for implementation use and its contents are not specified by ISO/IEC 13346.

14.10.8 Implementation Use Extended Attribute

The Implementation Use Extended Attribute shall be recorded in the format shown in figure 4/35.

RBP	Length	Name	Contents
0	4	Attribute Type	Uint32 (1/7.1.5) = 2 048
4	1	Attribute Subtype	Uint8 (1/7.1.1) = 1
5	3	Reserved	#00 bytes
8	4	Attribute Length	Uint32 (1/7.1.5)
12	4	Implementation Use Length (=IU_L)	Uint32 (1/7.1.5)
16	32	Implementation Identifier	regid (1/7.4)
48	IU_L	Implementation Use	bytes

Figure 35 - Implementation Use Extended Attribute format

14.10.8.1 Attribute Type (RBP 0)

This field shall specify 2 048.

14.10.8.2 Attribute Subtype (RBP 4)

This field shall specify 1. All other values are reserved for future standardisation.

14.10.8.3 Reserved (RBP 5)

This field shall be reserved for future standardisation and all bytes shall be set to #00.

14.10.8.4 Attribute Length (RBP 8)

This field shall specify the length of the entire extended attribute.

NOTE 39 - It is recommended that the extended attribute length be an integral multiple of 4.

14.10.8.5 Implementation Use Length (=IU_L) (RBP 12)

This field shall specify the length of the Implementation Use field.

14.10.8.6 Implementation Identifier (RBP 16)

This field shall specify an identification of an implementation which can recognise and act upon the contents of the Implementation Use field. If this field contains all #00 bytes, then no such implementation is identified. The scope of this regid includes the contents of the descriptors that specify the contents and attributes of the current file.

14.10.8.7 Implementation Use (RBP 48)

This field shall be reserved for implementation use. The interpretation of the contents of this field is not specified by this Part of ISO/IEC 13346.

14.10.9 Application Use Extended Attribute

The Application Use Extended Attribute shall be recorded in the format shown in figure 4/36.

RBP	Length	Name	Contents
0	4	Attribute Type	Uint32 (1/7.1.5) = 65 536
4	1	Attribute Subtype	Uint8 (1/7.1.1) = 1
5	3	Reserved	#00 bytes
8	4	Attribute Length	Uint32 (1/7.1.5)
12	4	Application Use Length(=AU_L)	Uint32 (1/7.1.5)
16	32	Application Identifier	regid (1/7.4)
48	AU_L	Application Use	bytes

Figure 36 - Application Use Extended Attribute format

14.10.9.1 Attribute Type (RBP 0)

This field shall specify 65 536.

14.10.9.2 Attribute Subtype (RBP 4)

This field shall specify 1. All other values are reserved for future standardisation.

14.10.9.3 Reserved (RBP 5)

This field shall be reserved for future standardisation and all bytes shall be set to #00.

14.10.9.4 Attribute Length (RBP 8)

This field shall specify the length of the entire extended attribute.

NOTE 40 - It is recommended that the extended attribute length be an integral multiple of 4.

14.10.9.5 Application Use Length(=AU_L) (RBP 12)

This field shall specify the length of the Application Use field.

14.10.9.6 Application Identifier (RBP 16)

This field shall specify an identification of an application which can recognise and act upon the contents of the Application Use field. If this field contains all #00 bytes, then no such application is identified. The scope of this `regid` includes the contents of the descriptors that specify the contents and attributes of the current file.

14.10.9.7 Application Use (RBP 48)

This field shall be reserved for application use. The interpretation of the contents of this field is not specified by this Part of ISO/IEC 13346.

14.11 Unallocated Space Entry

An Unallocated Space Entry is a direct entry recorded within an ICB and shall be recorded in the format shown in figure 4/37.

NOTE 41 - This is normally only used for write-once media.

BP	Length	Name	Contents
0	16	Descriptor Tag	`tag` (4/7.2) (Tag=263)
16	20	ICB Tag	`icbtag` (4/14.6) (Type=1)
36	4	Length of Allocation Descriptors (=L_AD)	`Uint32` (1/7.1.5)
40	L_AD	Allocation descriptors	bytes

Figure 37 - Unallocated Space Entry format

14.11.1 Descriptor Tag (BP 0)

The Tag Identifier field of the `tag` (4/7.2) for this descriptor shall contain 263.

14.11.2 ICB Tag (BP 16)

The File Type field of the `icbtag` (4/14.6) for this descriptor shall contain 1.

14.11.3 Length of Allocation Descriptors (=L_AD) (BP 36)

This field specifies the length, in bytes, of the Allocation Descriptors field. L_AD+40 shall not be greater than the size of a logical block.

14.11.4 Allocation Descriptors (BP 40)

This field shall contain allocation descriptors.

The type of allocation descriptor shall be specified by the Flags field in the ICB Tag field (see 4/14.6.8). The extent length fields of these allocation descriptors shall be an integral multiple of the logical block size.

14.12 Space Bitmap Descriptor

A Space Bitmap descriptor specifies a bit for every logical block in the partition and shall be recorded in the format shown in figure 4/38.

BP	Length	Name	Contents
0	16	Descriptor Tag	`tag` (4/7.2)(Tag=264)
16	4	Number of Bits (=N_BT)	`Uint32` (1/7.1.5)
20	4	Number of Bytes (=N_B)	`Uint32` (1/7.1.5)
24	N_B	Bitmap	bytes

Figure 38 - Space Bitmap Descriptor format

14.12.1 Descriptor Tag (BP 0)

The Tag Identifier field of the `tag` (4/7.2) for this descriptor shall contain 264.

14.12.2 Number of Bits (=N_BT) (BP 16)

This field shall specify the number of valid bits in the Bitmap field.

14.12.3 Number of Bytes (=N_B) (BP 20)

This field shall specify the number of bytes in the Bitmap field. The length of this field shall not be less than *ip((N_BT+7)/8)* bytes.

14.12.4 Bitmap (BP 24)

This field specifies a bit for each logical block in the partition. The bit for logical block *s* is bit *rem(s,8)* in byte *ip(s/8)*, where byte 0 is the first byte in this field.

14.13 Partition Integrity Entry

A Partition Integrity Entry is a direct entry recorded in an ICB and shall be recorded in the format shown in figure 4/39.

BP	Length	Name	Contents
0	16	Descriptor Tag	`tag` (4/7.2) (Tag=265)
16	20	ICB Tag	`icbtag` (4/14.6) (Type=2)
36	12	Recording Date and Time	`timestamp` (1/7.3)
48	1	Integrity Type	`Uint8` (1/7.1.1)
49	175	Reserved	#00 bytes
224	32	Implementation Identifier	`regid` (1/7.4)
256	256	Implementation Use	bytes

Figure 39 - Partition Integrity Entry format

14.13.1 Descriptor Tag (BP 0)

The Tag Identifier field of the `tag` (4/7.2) for this descriptor shall contain 265.

14.13.2 ICB Tag (BP 16)

The File Type field of the `icbtag` (4/14.6) for this descriptor shall contain 2.

14.13.3 Recording Date and Time (BP 36)

This field shall specify the date and time of the day of recording of this Integrity Entry.

14.13.4 Integrity Type (BP 48)

This field shall specify the type of Integrity Entry. The types are shown in figure 4/40.

Type	Interpretation
0	Shall mean that the entry is an Open Integrity Entry.
1	Shall mean that the entry is a Close Integrity Entry.
2	Shall mean that the entry is a Stable Integrity Entry.
3-255	Reserved for future standardisation.

Figure 40 - Integrity Entry interpretation

14.13.5 Reserved (BP 49)

This field shall be reserved for future standardisation and all bytes shall be set to #00.

14.13.6 Implementation Identifier (BP 224)

This field shall specify an identification of an implementation which can recognise and act upon the contents of the Implementation Use field. If this field contains all #00 bytes, then no such implementation is identified. The scope of this `regid` includes the contents of the partition associated with this descriptor.

14.13.7 Implementation Use (BP 256)

This field shall be reserved for implementation use. Its content is not specified by this Part.

14.14 Allocation descriptors

14.14.1 Short Allocation Descriptor

The Short Allocation Descriptor, designated as `short_ad`, shall be recorded in the format shown in figure 4/41.

RBP	Length	Name	Contents
0	4	Extent Length	`Uint32` (1/7.1.5)
4	4	Extent Position	`Uint32` (1/7.1.5)

Figure 41 - `short_ad` format

14.14.1.1 Extent Length (RBP 0)

The 30 least significant bits of this field shall be interpreted as a 30-bit unsigned binary number specifying the length of the extent in bytes. Unless otherwise specified, the length shall be an integral multiple of the logical block size. The 2 most significant bits shall be interpreted as a 2-bit unsigned binary number specifying the type of the extent as described in figure 4/42.

Value	Interpretation
0	Extent recorded and allocated
1	Extent not recorded but allocated
2	Extent not recorded and not allocated
3	The extent is the next extent of allocation descriptors (see 4/12)

Figure 42 - Extent interpretation

14.14.1.2 Extent Position (RBP 4)

This field shall specify the logical block number, within the partition the descriptor is recorded on, of the extent. If the extent's length is 0, no extent is specified and this field shall contain 0.

14.14.2 Long Allocation Descriptor

The Long Allocation Descriptor, designated by `long_ad`, shall be recorded in the format shown in figure 4/43.

RBP	Length	Name	Contents
0	4	Extent Length	`Uint32` (1/7.1.5)
4	6	Extent Location	`lb_addr` (4/7.1)
10	6	Implementation Use	bytes

Figure 43 - `long_ad` format

14.14.2.1 Extent Length (RBP 0)

This field shall be recorded as specified in 4/14.14.1.1.

14.14.2.2 Extent Location (RBP 4)

This field shall specify the logical block number of the extent. If the extent's length is 0, no extent is specified and this field shall contain 0.

14.14.2.3 Implementation Use (RBP 10)

This field shall be reserved for implementation use. Its content is not specified by this Part.

NOTE 42 - The `long_ad` (4/14.14.2) is intended for use when the extent's location may be on another partition (either on this volume or another).

14.14.3 Extended Allocation Descriptor

The Extended Allocation Descriptor, designated by `ext_ad`, shall be recorded in the format shown in figure 4/44.

RBP	Length	Name	Contents
0	4	Extent Length	`Uint32` (1/7.1.5)
4	4	Recorded Length	`Uint32` (1/7.1.5)
8	4	Information Length	`Uint32` (1/7.1.5)
12	6	Extent Location	`lb_addr` (4/7.1)
18	2	Implementation Use	bytes

Figure 44 - `ext_ad` format

14.14.3.1 Extent Length (RBP 0)

This field shall be recorded as specified in 4/14.14.1.1.

14.14.3.2 Recorded Length (RBP 4)

The two most significant bits of this field are reserved for future standardisation and shall be set to ZERO. The 30 least significant bits of this field shall be interpreted as a 30-bit unsigned binary number specifying the number of bytes recorded in the extent. This may be different from the number of bytes specified in the Extent Length field.

14.14.3.3 Information Length (RBP 8)

This field shall specify how many bytes of information are recorded starting at the first byte of the extent identified by the Extent Location field. This may be different from the value in either the Extent Length field or the Recorded Length field.

14.14.3.4 Extent Location (RBP 12)

This field shall specify the logical block number of the extent. If the extent's length is 0, no extent is specified and this field shall contain all #00 bytes.

14.14.3.5 Implementation Use (RBP 18)

This field shall be reserved for implementation use. Its content is not specified by Part.

NOTE 43 - The `ext_ad` (4/14.14.3) is similar to the `long_ad` (4/14.14.2) except that while Information Length bytes are represented in the extent, only Recorded Length bytes have been recorded. (Most likely, a compression algorithm has been applied on the extent.) The Recorded Length allows implementations to copy files (and their extents) without knowing how or why the Recorded Length differs from the Information Length.

14.15 Logical Volume Header Descriptor

The Logical Volume Header Descriptor shall specify a numeric file and directory identifier and shall be recorded with the format shown in figure 4/45 (see 4/3.1 for where this descriptor is recorded).

RBP	Length	Name	Contents
0	8	Unique Id	`Uint64` (1/7.1.7)
8	24	Reserved	#00 bytes

Figure 45 - Logical Volume Header Descriptor format

14.15.1 Unique Id (RBP 0)

This field shall specify a value which is greater than the value of the Unique Id field in any File Entry recorded on the associated logical volume.

NOTE 44 - The intended use of this field is to facilitate allocation of unique identifiers for files and directories (see 4/14.9.18). In order to avoid having to examine every file and directory, this field should be maintained even if the rest of the volume is not conformant with ISO/IEC 13346. An implementation might record

several Open Integrity Descriptors consecutively just to maintain this field as a modest increment over the last value recorded. Implementations should not assume any ordering properties for the value of this field; the value might decrease or increase with successive descriptors. In particular, the value in a Close Integrity Descriptor might be less than the value in the preceding Open Integrity Descriptor.

14.15.2 Reserved (RBP 8)

This field shall be reserved for future standardisation and all bytes shall contain #00.

14.16 Pathname

14.16.1 Path Component

A Path Component shall be recorded in the format shown in figure 4/46.

RBP	Length	Name	Contents
0	1	Component Type	Uint8 (1/7.1.1)
1	1	Length of Component Identifier (= L_CI)	Uint8 (1/7.1.1)
2	2	Component File Version Number	Uint16 (1/7.1.3)
4	L_CI	Component Identifier	d-characters (1/7.2)

Figure 46 - Path Component format

14.16.1.1 Component Type (RBP 0)

This field shall specify the component type as shown in figure 4/47.

Type	Interpretation
0	Reserved for future standardisation.
1	If L_CI is not 0, the component specifies the root of a directory hierarchy subject to agreement between the originator and recipient of the medium. If L_CI is 0, this component shall specify the root of a file system as specified in ISO/IEC 9945-1.
2	The component specifies the root directory of the directory hierarchy of which the predecessor of the first component in the pathname is a member.
3	The component specifies the parent directory of the predecessor component.
4	The component specifies the same directory as the predecessor component.
5	The component identifies an object, either a file or a directory or an alias, specified by a descriptor of the directory identified by the predecessor component, such that the contents of the File Identifier field of that directory descriptor is identical to the contents of the Component Identifier field.
6-255	Reserved for future standardisation.

Figure 47 - Component interpretation

14.16.1.2 Length of Component Identifier (= L_CI) (RBP 1)

If the Component Type field contains 1 or 5, this field shall specify the length in bytes of the Component Identifier field. If the Component Type field contains 5, L_CI shall be greater than 0. If the Component Type field does not contain 1 or 5, this field shall contain 0.

14.16.1.3 Component File Version Number (RBP 2)

This field shall specify the file version number of the component as follows.

If the number in this field is 0, then the highest file version number of any instance of the entity identified by the Component Identifier field (see 4/8.7) is identified.

If the number in this field is in the range 1 to 32 767 inclusive, this field shall specify the file version number of the entity identified by the Component Identifier field (see 4/8.7). The numbers 32 768 to 65 535 inclusive are reserved for future standardisation.

If the entity identified by the Component Identifier field (see 4/8.7) is a directory, then the value of this field shall be 0.

NOTE 45 - This allows versions of files and aliases to be specified in recorded pathnames.

14.16.1.4 Component Identifier (RBP 4)

This field shall identify the component.

15 Levels of medium interchange

This Part of ISO/IEC 13346 specifies three levels of medium interchange. The level of a file set shall be that level specifying the most restrictions required to record the file set according to the specifications of this Part of ISO/IEC 13346.

15.1 Level 1

At level 1, the following restrictions shall apply:

- The number in any Length of File Identifier field shall not exceed 12.
- A File Identifier (see 4/14.4) for a directory shall conform to the schema shown in figure 4/48. A sequence of fileid-characters shall be a sequence of d-characters (1/7.2) excluding SPACE, COMMA, FULL STOP and REVERSE SOLIDUS characters except as part of a code extension character (see 1/7.2.9.1).

```
[Directory File Identifier]{
      <fileid-characters>1+8
}
```

Figure 48 - Directory file identifier schema

- A File Identifier (see 4/14.4) for a non-directory file shall conform to the schema shown in figure 4/49.

```
[Nondirectory File Identifier]{
      <fileid-characters>1+8
   | {
            <fileid-characters>1+8
            <FULL STOP character>
            <fileid-characters>0+3
     }
   | {
            <fileid-characters>0+8
            <FULL STOP character>
            <fileid-characters>1+3
     }
}
```

Figure 49 - Nondirectory file identifier schema

- There shall not be more than one descriptor in a directory with the same File Identifier.
- The value of the File Link Count field in a File Entry shall not exceed 8.
- No File Entries representing symbolic links shall be recorded.
- The maximum length of a resolved pathname (4/8.7.1) shall not exceed 64.

NOTE 46 - For many systems, there are certain file identifiers which will cause problems during interchange. For maximum interchange, the following file identifiers should not be used

AUX CLOCK$ COM*n* CON LPT*m* NUL PRN

where ***n*** is one of the four characters DIGITs ONE to FOUR and ***m*** is one of the three characters DIGITs ONE to THREE.

NOTE 47 - The restriction on the maximum size of resolved pathnames may be difficult to enforce incrementally. For example, changing a directory 's name requires, in principle, checking all pathnames including that directory. It may be simpler to check this restriction as a separate processing step prior to interchange.

15.2 Level 2

At Level 2, the following restrictions shall apply:

- The number in any Length of File Identifier and Length of Component Identifier field shall not exceed 14.