INTERNATIONAL STANDARD

ISO 8571-2

First edition
1988-10-01

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION
ORGANISATION INTERNATIONALE DE NORMALISATION
МЕЖДУНАРОДНАЯ ОРГАНИЗАЦИЯ ПО СТАНДАРТИЗАЦИИ

Information processing systems — Open Systems Interconnection — File Transfer, Access and Management —

Part 2 : Virtual Filestore Definition

Systèmes de traitement de l'information — Interconnexion de systèmes ouverts — Gestion, accès et transfert de fichier —

Partie 2 : Fichier virtuel

Reference number
ISO 8571-2 : 1988 (E)

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

Draft International Standards adopted by the technical committees are circulated to the member bodies for approval before their acceptance as International Standards by the ISO Council. They are approved in accordance with ISO procedures requiring at least 75 % approval by the member bodies voting.

International Standard ISO 8571-2 was prepared by Technical Committee ISO/TC 97, *Information processing systems.*

Users should note that all International Standards undergo revision from time to time and that any reference made herein to any other International Standard implies its latest edition, unless otherwise stated.

ISO 8571 consists of the following parts, under the general title *Information processing systems — Open Systems Interconnection — File Transfer, Access and Management*

- *Part 1 : General introduction*
- *Part 2 : Virtual Filestore Definition*
- *Part 3 : File Service Definition*
- *Part 4 : File Protocol Specification*

Annexes A and B form an integral part of this International Standard. Annexes C, D and E are for information only.

Printed in Switzerland

Contents

Annexes

Figures

Tables

This page intentionally left blank

Information processing systems — Open Systems Interconnection — File Transfer, Access and Management —

Part 2 : Virtual Filestore Definition

0 Introduction

ISO 8571 is one of a set of International Standards produced to facilitate the interconnection of computer systems. It is related to other International Standards in the set as defined by the Reference Model for Open Systems Interconnection (ISO 7498). The Reference Model subdivides the area of standardization for interconnection into a series of layers of specification, each of manageable size.

The aim of Open Systems Interconnection is to allow, with a minimum of technical agreement outside the interconnection standards, the interconnection of computer systems

a) from different manufacturers,

b) under different managements,

c) of different levels of complexity,

d) of different ages.

ISO 8571 defines a File Service and specifies a File Protocol available within the application layer of the Reference Model. The service defined is of the category Application Service Element (ASE). It is concerned with identifiable bodies of information which can be treated as files, and may be stored within open systems or passed between application processes.

ISO 8571 defines a basic file service. It provides basic facilities to support file transfer, and establishes a framework for file access and file management. ISO 8571 does not specify the interfaces to a file transfer or access facility within the local system.

ISO 8571 consists of the following four parts:

Part 1: General introduction
Part 2: Virtual Filestore definition
Part 3: File Service definition
Part 4: File Protocol specification

The definition in this part of ISO 8571 is used in the subsequent parts of ISO 8571 which specify services and protocols. They reference the filestore definition in order to assign meaning to the various descriptive data items which they manipulate. This definition will also be used by protocol implementors when choosing a mapping from the protocol items onto their real storage mechanism.

This part of ISO 8571 contains the following annexes which form part of the standard:

Annex A - File access structure constraint sets;
Annex B - Document types;

and the following annexes which do not form part of the standard:

Annex C - Reading of structured files;
Annex D - Insertion in a structured file;
Annex E - ASN.1 cross reference.

1 Scope and field of application

This part of ISO 8571

a) defines an abstract model of the virtual filestore for describing files and filestores (see section one);

b) defines the set of actions available to manipulate the elements of the model (see section two);

c) defines the properties of individual files and associations in terms of attributes (see section three);

d) defines the form of representations of files with hierarchical structures (see clause 7 in section one).

This part of ISO 8571 does not specify

e) requirements relating to the mapping from real to virtual filestores;

f) requirements for implementations of the real filestore.

The Virtual Filestore definition is provided for reference by the other parts of ISO 8571 defining the file service (ISO 8571-3) and specifying the file protocol (ISO 8571-4).

2 References

ISO 6429, *Information processing - ISO 7-bit and 8-bit coded character sets - Additional control functions for character imaging devices.*

ISO 7498, *Information Processing Systems - Open Systems Interconnection - Basic Reference Model.*

ISO 8571, *Information Processing Systems - Open systems Interconnection - File transfer, access and management.*
- *Part 1: General introduction.*
- *Part 3: File Service definition.*
- *Part 4: File Protocol specification.*

ISO 8601, *Data elements and interchange formats - Information interchange - Representation of dates and times.*

ISO 8650, *Information Processing Systems - Open Systems Interconnection - Protocol specification for the Association Control Service Element.*

ISO 8822, *Information Processing Systems - Open Systems Interconnection - Connection-oriented presentation service definition.*

ISO 8824, *Information Processing Systems - Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1).*

ISO 8825, *Information Processing Systems - Open Systems Interconnection - Specification of Basic encoding rules for Abstract Syntax Notation One (ASN.1).*

ISO 8832, *Information Processing Systems - Open Systems Interconnection - Specification of the basic class protocol for Job Transfer and Manipulation.* [1)]

ISO 9804, *Information Processing Systems - Open Systems Interconnection - Definition of Application Service Elements - Commitment, Concurrency and Recovery.* [1)]

ISO 9834-2, *Information Processing Systems - Procedures for specific OSI registration authorities - Part 2: Registration of Document Types.* [1)]

3 Definitions

Terms are defined in ISO 8571-1.

4 Abbreviations

Abbreviations are defined in ISO 8571-1.

1) At present at the stage of draft; publication anticipated in due course.

Section one: The filestore model

5 Basic concepts

A virtual filestore is handled by one or more application entities; an initiator may communicate with these by initializing an FTAM regime, in which the file service is provided. Each application entity that is capable of operating as a responder in an FTAM regime supports a single virtual filestore. Thus an application entity title serves to identify a virtual filestore.

NOTE - Mechanisms for allocation of unambiguous application entity titles and for the corresponding directory services to provide addresses for communication with the entity are outside the scope of ISO 8571.

A filestore may contain an arbitrary number (greater than or equal to zero) of files (see figure 1).

The properties of each file are defined by the values of a set of file attributes. These attributes are global; at any one time, a single attribute value is available to all initiators.

Each file is either empty or has some contents and a structure. Some of the file attributes identify structural aspects of the contents.

There is a set of activity attributes associated with each FTAM regime. The activity attributes are of two types.

The first are in one to one correspondence with the file attributes, and indicate the active value of those attributes as perceived by the initiator.

The second type of activity attributes are current values of state information concerning the exchanges taking place within an FTAM regime, and the facts established about the initiator. These are usually derived from corresponding parameters in protocol exchanges. An initiator has available the description of only the FTAM regimes in which it is participating.

An arbitrary number (greater than or equal to zero) of initiators may have initialized FTAM regimes at any one

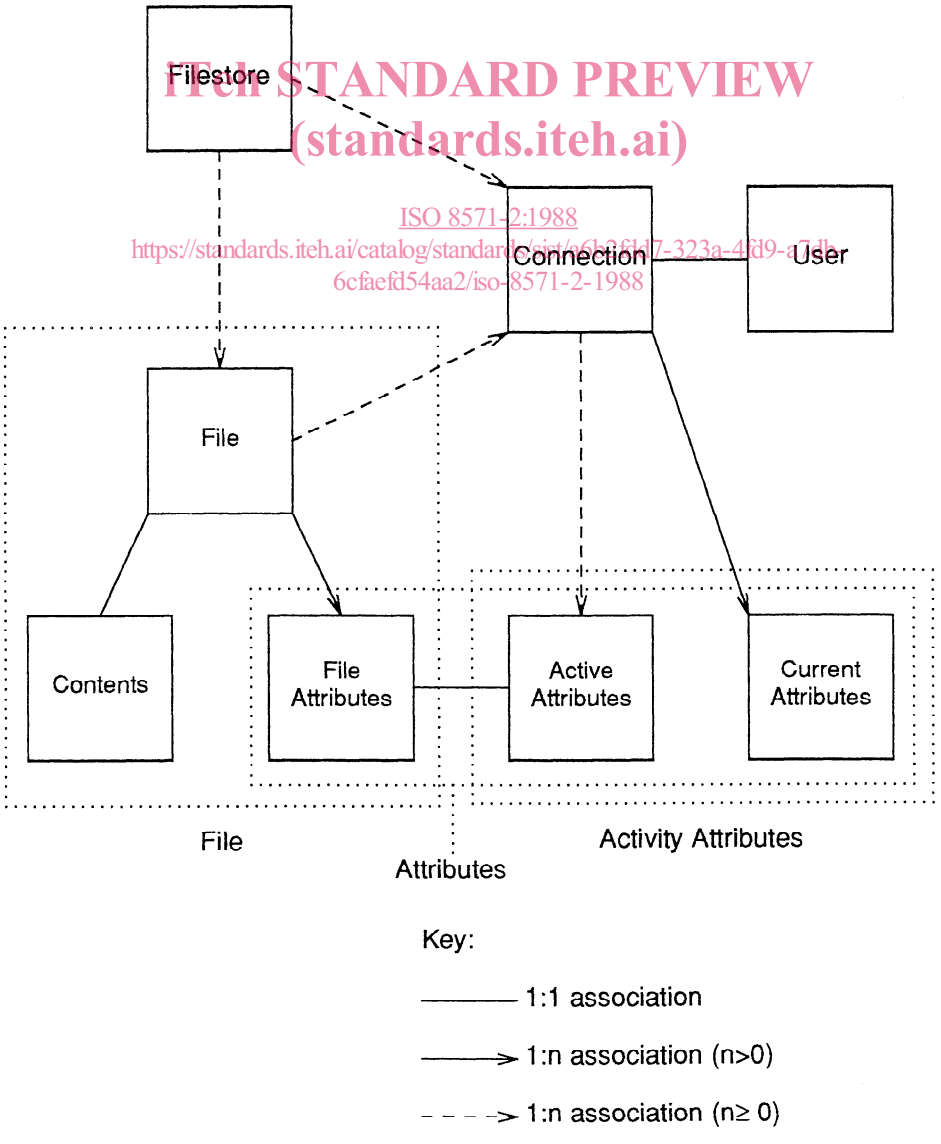


Figure 1 — Relationship of files, attributes and associations

time. Exchanges between the initiator and the responder lead to the selection of at most one file in the responder's virtual filestore to be bound to a particular FTAM regime at any one time.

6 File selection

From outside the filestore, selection of a file is always made by reference to the name of a file. The reference to a file is within the context of a particular filestore identified by the application entity title. The application entity title refers to the location of file storage, and is known to the file service users, but lies outside the scope of FTAM. The filename is defined in clause 12.

Selection of a file takes place in two stages. First, an FTAM regime is initialized with the application entity handling the virtual filestore, and then information is given to this entity, to identify the file unambiguously from amongst all the files in the filestore.

In general, selection could be made by the statement of a number of relations between given values and file attributes. However, in ISO 8571 reference to the file is always made in terms of the filename.

7 File structures

7.1 File access structure

This clause defines the properties of a hierarchical structure. The abstract structure of hierarchical files, including the simple cases of flat and unstructured files, is defined in this part using ASN.1 in the ASN.1 module ISO8571-FADU (see 7.2). A hierarchical structure has the following properties:

a) The file access structure is an ordered tree.

b) Zero or one Data Units are assigned to a node.

c) Each node within the structure gives access to its subtree. The access unit (that is, the subtree) is known as a File Access Data Unit (FADU) and it is comprised of the structural nodes of the subtree and the Data Units contained within the subtree. The root node of the tree gives access to the entire file.

d) Optionally a node has a name assigned to it.

e) The number of levels, the arc lengths and the number of arcs originating from each node are not restricted.

NOTES

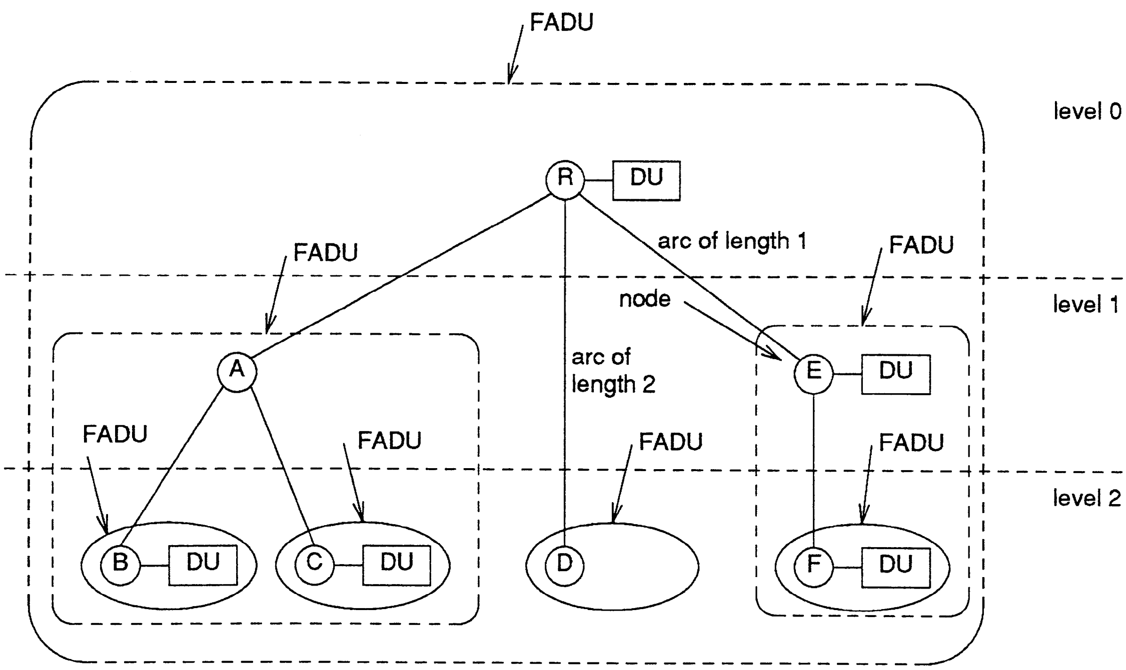
1 Applicable constraint sets may limit the maximum level, and the arc lengths allowed.

2 These terms are illustrated in the figure 2, where, for tutorial purposes, each node has been uniquely identified by the letters A to F.

A preorder traversal sequence is a specific sequence of nodes in the tree; it is established by traversing the subtree corresponding to the whole file. A subtree is traversed by appending the root node of the subtree to the end of the traversal sequence established so far, and then, for each of the children of the subtree in their order of appearance, traversing that subtree.

NOTE - The algorithm describes the abstract construction of a sequence, which is used for establishing FADU Identity, defining the location operations and determining the order of transmission. This part of ISO 8571 does not define how the algorithm is to be implemented.

The File Access Data Units (FADUs), being in one-to-one correspondence with the subtrees, can be identified in the same manner as subtrees (that is, by their root nodes). Likewise, data units are always associated with a node and may be identified by the identification of the node. The general hierarchical structure can represent a wide range of different practical file structures. However, real systems can



NOTE - the following features of the tree are identified (subtrees are identified by their root node):
Root node of tree : node R
Preorder traversal sequence : nodes R, A, B, C, D, E, F
Order of appearance : left to right

Figure 2 — The Access Structure as a Tree Structure

only support a limited range of structures, and there are restrictions on the way files can be modified. To express this, the concept of a constraint set is introduced. A constraint set defines limitations on the range of structures allowed, and expresses the way in which the basic access actions can modify the structure. Constraint sets reflecting certain common file types are defined in this part of ISO 8571 (see annex A), but other constraint sets may subsequently be defined and registered.

NOTE - It is expected that a registration authority will be established to maintain a register of constraint sets.

7.2 Abstract structure definition

The access structure of the hierarchical file model is specified in figure 4, using ASN.1. The data unit contents may be expressed in ASN.1, as in figure 3, or in some other abstract syntax notation.

Files are accessed in terms of File Access Data Units (FADU), which are equivalent to the subtrees in the general hierarchical model. The smallest amount of data to be specified for access is one Data Unit (DU). For the purpose of transfer with checkpointing, the Data Units may be divided into smaller parts called data elements (DE). It is not possible to access the individual data elements of a Data Unit by means of the filestore actions defined in ISO 8571; the data unit is accessed as a whole.

7.3 Abstract syntax definition

For the purpose of providing access to the structure of files in the FTAM hierarchical file model this International Standard assigns the ASN.1 object identifier value

{ iso standard 8571 abstract-syntax (2) ftam-fadu (2) }

as an abstract syntax name for the set of presentation data values, each of which is a value of the ASN.1 type ISO8571-FADU.Structuring-Data-Element.

The corresponding object descriptor value shall be

"FTAM FADU"

The ASN.1 object identifier and object descriptor values

{ joint-iso-ccitt asn1 (1) basic-encoding (1) }

and

"Basic Encoding of a single ASN.1 type"

(assigned to an information object in ISO 8825) can be used as a transfer syntax name with this abstract syntax.

7.4 File transfer structure

The ASN.1 module ISO8571-FADU defines the access structure of the general hierarchical model. The syntax for transfer of files' contents is derived from this and consists of values of type Data-Element. Only values of type Data-Element are transferred from the FTAM sender to the FTAM receiver by means of the Presentation Service, ISO 8822. However these values shall appear in their syntactical order.

NOTE - The syntax is constructed so that the syntactical order is equivalent to the preorder traversal sequence.

Structuring information (i.e. values of type Structuring-Data-Element) is communicated in FTAM FADU context (the one corresponding to the FTAM FADU abstract syntax, see 7.3). Node names are either in the same context or in a different, but embedded context. Values of type File-Contents-Data-Element are communicated in the file contents presentation context.

The abstract syntaxes used to transfer part or all of the contents of a file are indicated by the contents type file attribute. Each abstract syntax used shall be associated with a different presentation context. There are two possibilities:

a) The contents type file attribute value specifies an abstract syntax and a constraint set; a presentation context is required corresponding to the specified abstract syntax. This presentation context is used to transfer both Node-Names and the actual contents of the file. A distinct presentation context, which corresponds to the file structuring abstract syntax (see 7.3), is required if the constraint set in use supports file structuring information.

b) The contents type file attribute value specifies a document type; a presentation context is required corresponding to each abstract syntax specified in the document type register entry. The document type also defines which presentation context is to be used for transfer of Node-Names, if present. They can be

1) in a presentation context corresponding to one of the abstract syntaxes defined by the document type (user-coded);

2) in the same presentation context as the file structuring information (ftam-coded).

7.5 Access context

Use of the abstract structure defined in the ISO8571-FADU module to derive the corresponding sequence of information for transfer will yield the full hierarchical structure of the files, i.e. all the structuring information and all the data in the specified FADU will be transferred. However it is possible to access files for reading with a restricted view of their structure by use of different access contexts. In all cases, those data elements which are transferred are in the order defined in ISO8571-FADU, and nodes are transferred in the order in which they appear in the preorder traversal sequence.

```
ISO8571-CONTENTS DEFINITIONS ::=

BEGIN

  File-Contents-Data-Element ::= ANY
    -- Values of File-Contents-Data-Element are always transferred in a
    -- presentation context which is different from the presentation context
    -- used to transfer FTAM PCI. The actual presentation data valuesallowed
    -- are found in the abstract syntax for the file contents, as specified
    -- in the contents type file attribute for the file.

END
```

Figure 3 — ASN.1 definition of file contents

```
ISO8571-FADU DEFINITIONS ::=

BEGIN

  Subtree ::= SEQUENCE {
    node          Node-Descriptor-Data-Element,
    data          [0] IMPLICIT DU OPTIONAL,
      -- present if and only if a DU is connected to the node.
    children      [1] IMPLICIT Children OPTIONAL }
      -- a leaf node is characterised by having no children

  Children ::= SEQUENCE {
    enter-subtree    Enter-Subtree-Data-Element,
    SEQUENCE OF Subtree,
      -- subtrees must appear in their proper order according to
      -- their proper ordering as children of their parent node.
    exit-subtree     Exit-Subtree-Data-Element }

  DU ::= SEQUENCE OF ISO8571-CONTENTS.File-Contents-Data-Element

  Node-Descriptor-Data-Element ::= [APPLICATION 0] IMPLICIT SEQUENCE {
    name             Node-Name OPTIONAL,
      -- present only if the root node of the subtree is a named node.
    arc-length       [1] IMPLICIT INTEGER DEFAULT 1,
      -- used to specify the length of the arc to the root node of the subtree
      -- from its parent node.
    data-exists      [2] IMPLICIT BOOLEAN DEFAULT TRUE }
      -- data-exists = TRUE indicates that a DU is connected to the root node
      -- of the subtree.

  Node-Name ::= CHOICE {
    ftam-coded       [0] IMPLICIT GraphicString,
      -- when ftam-coded is used, the Node-Name belongs to the same abstract syntax
      -- as the structuring information. Node-Names are then transferred in
      -- the presentation context established to support the FTAM FADU abstract
      -- syntax. This form of Node-Name is only allowed when the contents type
      -- file attribute contains a document type name. To support this
      -- alternative, at least the G0 character set registered in character
      -- set register entry 2 shall be supported.
    user-coded       EXTERNAL }
      -- the actual types allowed are found in the abstract syntax for the files
      -- contents, as specified in the contents type file attribute for the file.

    Enter-Subtree-Data-Element ::= [APPLICATION 1] IMPLICIT NULL

    Exit-Subtree-Data-Element ::= [APPLICATION 2] IMPLICIT NULL
      -- the enter-subtree and exit-subtree data elements are used to bracket
      -- the list of subtrees, which are children of the preceding node.

    FADU ::= Subtree

    Structuring-Data-Element ::= CHOICE {
      Node-Descriptor-Data-Element,
      Enter-Subtree-Data-Element,
      Exit-Subtree-Data-Element }

    -- Data-Element is defined to be a general data type whose values are
    --
    --   a) a value of the ASN.1 type Structuring-Data-Element in the abstract
    --      syntax "FTAM FADU"; or
    --
    --   b) a value of the ASN.1 type ISO8571-CONTENTS.File-Contents-Data-Element
    --      in the abstract syntax derived from the contents type file attribute.
END
```


Figure 4 — ASN.1 definition of file structure

7.5.1 HA - Hierarchical All Data Units Access Context

In the HA access context, all four types of data elments (Node-Descriptor-Data-Element, Enter-Subtree-Data-Element, Exit-Subtree-Data-Element and File-Contents-Data-Element) in the addressed FADU are transferred.

7.5.2 HN - Hierarchical No Data Units Access Context

In the HN access context, all data elements of the types Node-Descriptor-Data-Element, Enter-Subtree-Data-Element and Exit-Subtree-Data-Element from the addressed FADU are transferred.

7.5.3 FA - Flat All Data Units Access Context

In the FA access context, data elements of the types Node-Descriptor-Data-Element and File-Contents-Data-Element from the addressed FADU are transferred. Only those Node-Descriptor-Data-Elements in which data-exists has the value TRUE are transferred.

7.5.4 FL - Flat One Level Data Units Access Context

In the FL access context, data elements of the types Node-Descriptor-Data-Element and File-Contents-Data-Element from those nodes in the addressed FADU which belongs to the specified level are transferred. Only those Node-Descriptor-Data-Elements in which data-exists has the value TRUE are transferred.

7.5.5 FS - Flat Single Data Unit Access Context

In the FS access context, the single Node-Descriptor-Data-Element and all the File-Contents-Data-Elements of the single DU belonging to the root node of the addressed FADU are transferred.

7.5.6 UA - Unstructured All Data Units Access Context

In the UA access context, only the data elements of type File-Contents-Data-Element from the addressed FADU are transferred.

7.5.7 US - Unstructured Single Data Unit Access Context

In the US access context, all the data elements of type File-Contents-Data-Element of the single DU belonging to the root node of the addressed FADU are transferred.

Table 1 — Result of reading in an access context

Access Context	Result
HA	Single Subtree
HN	undefined type - because Node-Descriptor-Data-Elements with data exists = TRUE are transferred without the data elements constituting the DU.
FA	series of Subtree each with one node
FL	series of Subtree each with one node
FS	Single Subtree with one node
UA	Single DU
US	Single DU

7.5.8 Summary of access contexts

Reading a non empty hierarchical subtree in these access contexts will have the results shown in table 1, in terms of the data types defined in ISO8571-FADU.

7.6 Identification structure

A FADU is identified by referencing the root node of the corresponding subtree. A FADU can be identified by any of the following mechanisms:

a) "first": the first FADU in the preorder traversal sequence for the file structure for which data exists is identified;

b) "last": the last FADU in the preorder traversal sequence for the file structure is identified;

c) "previous": the FADU preceding the currently identified FADU in the preorder traversal sequence of the file access structure is identified;

d) "current": the current location (see clause 8) in the file remains unchanged;

e) "next": the FADU following the currently identified FADU in the preorder traversal sequence of the file access structure is identified;

f) "begin": the exact meaning of begin depends on the constraint set in use, but is such that "locate next" will identify the "first" FADU.

g) "end": establishes a state of the file where there is no current location, but use of "previous" will identify the "last" FADU in the preorder traversal sequence for the file access structure;

h) Node-Name: specifies the identifier of the FADU which is to be identified. The search for the specific Node-Name is restricted to the children of the currently located node;

i) Sequence of Node-Names: specifies a path of FADU identifiers from the root node of the file to the node to be located. The first Node-Name is a child of the root node of the file, so that the root node itself is identified by an empty sequence of Node-Names;

j) Node number: specifies the node to be selected by its number in the preorder traversal sequence for the file access structure. The root node of the file has node number zero;

7.7 Constraint sets

The general hierarchical structure is constrained and its dynamics further qualified by the definition of a number of constraint sets, each identifying a particular structure in common use. These are specified in annex A. Constraint sets are referenced in protocol exchanges by names, which are of ASN.1 type "object identifier".

The set of actions that are ultimately allowable in the open regime is actually further constrained beyond the actions specified in the constraint set. This further restriction is imposed by the permitted actions file attribute, the processing mode activity attribute, the access control file attribute, the concurrency control activity attribute and the functional units selected for the current FTAM regime.

8 Actions on files

The virtual filestore defines actions which manipulate the data units in a file. The definition of the individual actions (see section two) states the data units to which the actions apply, and the effects on those data units. Some actions also establish filestore state, such as the state "file selected" or FADU location.

The actions are invoked by service primitives. Their semantics are defined in conjunction with those of the file service primitives defined in ISO 8571-3 (see note 1).