INTERNATIONAL TELECOMMUNICATION UNION

CCITT

T.434

THE INTERNATIONAL
TELEGRAPH AND TELEPHONE
CONSULTATIVE COMMITTEE

(09/92)

TERMINAL EQUIPMENT AND PROTOCOLS FOR TELEMATIC SERVICES

BINARY FILE TRANSFER FORMAT FOR THE TELEMATIC SERVICES

Recommendation T.434

FOREWORD

The CCITT (the International Telegraph and Telephone Consultative Committee) is a permanent organ of the International Telecommunication Union (ITU). CCITT is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The Plenary Assembly of CCITT which meets every four years, establishes the topics for study and approves Recommendations prepared by its Study Groups. The approval of Recommendations by the members of CCITT between Plenary Assemblies is covered by the procedure laid down in CCITT Resolution No. 2 (Melbourne, 1988).

Recommendation T.434 was prepared by Study Group VIII and was approved under the Resolution No. 2 procedure on the 18th of September 1992.

____

CCITT NOTE

In this Recommendation, the expression “Administration” is used for conciseness to indicate both a telecommunication administration and a recognized private operating agency.

CONTENTS

INTRODUCTION

Recommendation T.434 is one of the Recommendations of the T.430-Series Document Transfer and Manipulation (DTAM) that contain the protocol definitions and service descriptions used in the transfer of documents and data in the Telematic services.

Recommendation T.434

BINARY FILE TRANSFER FORMAT FOR THE TELEMATIC SERVICES

(1992)

1 Scope

This Recommendation defines the Binary File Transfer format which is intended for the transfer of data in the Telematic services, including Teletex Transparent Mode, Telefax 3 and 4, DTAM, and Message Handling.

2 Normative references

References are contained in Recommendations T.431 and T.400.

The following International Standards are referenced in this Recommendation in addition to the references of Recommendations T.431 and T.400. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this Recommendation are encouraged to investigate the possibility of applying the most recent edition of the Recommendations and Standards listed below.

- ISO 2014, *Writing of Calendar Dates in All-numeric Form.*

- ISO 3307, *Information Interchange – Representation of Time of the Day Version.*

- ISO 4031, *Information Interchange – Representation of Local Time Differentials.*

- ISO 8571-2, *Information Processing Systems – Open Systems Interconnection – File Transfer, Access and Management (FTAM) – Part 2: Virtual Filestore Definitions.*

- ISO 8571-4, *Information Processing Systems – Open Systems Interconnection – File Transfer, Access and Management (FTAM) - Part 4: File Protocol Specification.*

3 Definitions

3.1 attribute

A piece of information stating a property of something, taking one of a set of defined values, each value having a defined meaning.

3.2 binary file (data)

A sequence of octets, representing a binary file and optional attributes, formed, using the coding rules in Appendix A.

3.3 file attributes

The name and other identifiable properties of a file.

4 General concept

Binary File Transfer (BFT) describes the semantics and syntax necessary to represent a data file in order to transfer it through the protocols of various Telematic services, such as Teletex, Telefax Group 3 and Group 4, DTAM normal mode and Message Handling.

The following sections describe the attributes of a data file.

5 BFT file attributes

For the BFT, the following attributes (see Table 1/T.434) are defined. All of the attributes are optional. They are described using ASN.1.

A BINARY DATA message consists of a sequence of these attributes which include the file data itself and is defined as follows.

```
BINARY-DATA-Message             ::= [APPLICATION 23] IMPLICIT SEQUENCE OF
                                {SEQUENCE { [...list of attributes...] } }
```

The following sections contain the syntax for each attribute.

5.1 *File name syntax*

The file name is a sequence of name components. Each component is a value of type GraphicString.

When more than one element is encoded, the first element shall be the file name and the remaining elements shall be concatenated to represent the file name prefix.

```
filename                        [0] IMPLICIT Filename-Attribute OPTIONAL,
Filename-Attribute              ::= SEQUENCE OF GraphicString
```

TABLE 1/T.434

File attributes

Attribute Name
protocol version
filename
permitted actions
contents type
storage account
date and time of creation
date and time of last modification
date and time of last read access
identity of creator
identity of last modifier
identity of last reader
filesize
future filesize
access control
legal qualifications
private use
structure
application reference
machine
operating system
recipient
character set
compression
environment
pathname
user visible string
data file content

5.2 *Permitted actions syntax*

The permitted actions attribute indicates the set of actions that can be performed on the file.

```
permitted-actions                 [1] IMPLICIT Permitted-Actions-Attribute OPTIONAL,

Permitted-Actions-Attribute       ::= BIT STRING
    -- Actions available {

       read       (0),
       insert     (1),
       replace    (2),
       extend     (3),
       erase      (4) } (size (2...2))
```

5.3 *Contents Type Syntax*

The contents type attribute indicates the abstract data types of the contents of the file and the structuring information which is necessary if the complete file structure and semantics are to be maintained during the transfer of the file.

The value is either a document name (possibly with parameters in a single value of any type) or a pair of abstract syntax name and constraint set name. Each of these names are values of the type OBJECT IDENTIFIER.

```
contents-type                           [2] Contents-Type-Attribute OPTIONAL,
Contents-Type-attribute                 ::=  {

        document-type                   [0] IMPLICIT SEQUENCE {
            document-type-name          [1] Document-Type-Name,
            parameter                   [0] ANY OPTIONAL }

            -- The actual types to be used for values of the parameter
            -- field are defined in the named document type.
            -- Currently, only UNSTRUCTURED TEXT and UNSTRUCTURED BINARY
            -- are supported, UNSTRUCTURED BINARY is the default value.

Document-Type-Name                      ::= OBJECT IDENTIFIER
```

5.4 *Storage account syntax*

The storage account attribute identifies the accountable authority responsible for accumulated file storage charges. The value of the storage account attribute is of type GraphicString.

```
storage-account                         [3] IMPLICIT GraphicString OPTIONAL,
```

5.5 *Date and time syntax*

The date and time of creation attribute indicates when the file was created. The value of the attribute is of type GeneralizedTime.

The date and time of last modification attribute indicates when the contents of the file were last modified. The value of the attribute is of type GeneralizedTime. For a newly created file, the value is equal to the value of the date and time of creation attribute.

The date and time of last read access attribute indicates when the contents of the file were last read. The value of the attribute is of type GeneralizedTime. For a newly created file, the value is equal to the value of the date and time of creation attribute.

```
date-and-time-of-creation               [4] IMPLICIT GeneralizedTime OPTIONAL
date-and-time-of-last-modification      [5] IMPLICIT GeneralizedTime OPTIONAL
date-and-time-of-last-read-access       [6] IMPLICIT GeneralizedTime OPTIONAL
```

Generalized Time represents a calendar date and time of day to various precisions, as provided for by ISO 2014, ISO 3307, and ISO 4031. The time of day can be specified as local time only, UTC time only, or as both local and UTC time.

The Generalized Time type is formally defined as shown below. It is a string of characters, as follows:

(1) Where the local time only is present, the Generalized Time is a string consisting of the date, as specified in ISO 2014, followed by the local time of day, using one of the forms specified in ISO 3307.

(2) Where the UTC time only is present, the representation is as for Case (1), followed by the letter “Z” to indicate that the time is based on UTC.

(3) Where both local time and UTC are present, the representation is as for Case (1), followed by a TDF (Time Differential Factor), as defined in ISO 4031, which represents the difference of local time for UTC.

The characters required to represent the Generalized Time (the digits “0” to “9”, “.”, “,”, “+”, “–”, and “Z”) are taken from International Alphabet Number 5:

```
GeneralizedTime                  ::= [UNIVERSAL 24] IMPLICIT IA5String
```

Example – If of type GeneralizedTime, the value “8201020700”, which represents a local time of 7 AM on 2 January 1982, can be encoded as (using hexadecimal notation):

Generalized Time	Length	Contents
18	08	38323031303230373030

5.6 *Identity attribute syntax*

The value of the identity of creator attribute is a GraphicString.

The identity of last modifier attribute is altered by the receiver whenever the file has been opened for modification or extension and is closed (including closure following a connection failure). The value is of type GraphicString. For a newly created file, the value is equal to the value of the identity of creator attribute.

The identity of last reader attribute is altered by the receiver whenever the file has been opened for reading and is closed (including closure following a connection failure). The attribute is of type GraphicString. For a newly created file, the value is equal to the value of the identity of creator attribute.

```
identity-of-creator              [8] IMPLICIT GraphicString OPTIONAL
identity-of-last-modifier        [9] IMPLICIT GraphicString OPTIONAL
identity-of-last-reader          [10] IMPLICIT GraphicString OPTIONAL
```

5.7 *File characteristic attributes syntax*

The filesize attribute is altered by the receiver whenever the file is closed after having been opened for modification and extension (including closure following a connection failure).

The attribute is set to the nominal size in octets of the complete file when the file is closed. The value of the attribute is an integer. For a newly created file, the value is set to zero.

The future filesize attribute indicates the nominal size in octets to which the file may grow as a result of modification and extension. The value of the attribute is an integer.

```
filesize                         [13] IMPLICIT INTEGER OPTIONAL
future-filesize                  [14] IMPLICIT INTEGER OPTIONAL
```

5.8 *Legal qualifications syntax*

The legal qualifications attribute conveys information about the legal status of the file and its use. The value of the attribute is of type GraphicString.

```
legal-qualifications             [16] IMPLICIT GraphicString OPTIONAL
```

5.9 *Private use syntax*

The meaning of the private use attribute is not defined. The value of the attribute can take any form.

```
private-use                      [17] Private-Use-Attribute OPTIONAL

Private-Use-Attribute            :: = SEQUENCE {
      manufacturer-values        [0] EXTERNAL OPTIONAL }
```

5.10 *Environment Syntax*

These attributes provide additional information describing various aspects of the environment the Binary File Transfer is originating from.

```
application-reference        [19] IMPLICIT SEQUENCE OF GraphicString OPTIONAL
machine                      [20] IMPLICIT SEQUENCE OF GraphicString OPTIONAL
operating-system             [21] IMPLICIT OBJECT IDENTIFIER OPTIONAL
environment                  [25] IMPLICIT SEQUENCE OF GraphicString OPTIONAL
pathname                     [26] IMPLICIT SEQUENCE OF GraphicString OPTIONAL
user-visible-string          [29] IMPLICIT SEQUENCE OF GraphicString OPTIONAL
```

Note – Application reference is intended for identifying application program and version numbers.

5.11 *Structure Syntax*

The structure attribute indicates the format of the data being transferred in the data-file-content attribute.

```
structure                    [18] IMPLICIT OBJECT IDENTIFIER OPTIONAL
```

5.12 *Recipient Syntax*

The recipient attribute is used to indicate the final user destination of the Binary File Transfer.

```
recipient                    [22] IMPLICIT SEQUENCE OF GraphicString OPTIONAL
```

5.13 *Character Set Syntax*

The character set attribute indicates the international character set to be used for the rendering of the character data contained in the attribute data-file-content.

```
character-set                [23] IMPLICIT OBJECT IDENTIFIER OPTIONAL
```

5.14 *Compression Syntax*

The compression attribute indicates an optional compression added to the contents of the data-file-content attribute.

```
compression                  [24] IMPLICIT SEQUENCE OF GraphicString OPTIONAL
```

5.15 *Data Format Syntax*

The data file content attribute contains the data file contents to be transferred.

```
data-file-content            [30] EXTERNAL OPTIONAL
```

5.16 *Access Control Syntax (for further study)*

The access control attribute is a set attribute. It defines conditions under which access to the file is valid.

Each element of the set gives one condition under which access to the file is valid. Access to the file is allowed if at least one of these conditions is satisfied. However, the access must be based on a single condition. and not on the union of a number of separate conditions.

```
access-control               [15] Access-Control-Attribute OPTIONAL

Access-Control-Attribute     ::= CHOICE {
    simple-password          [0] IMPLICIT OCTET STRING,
            -- A simplified form of the access control syntax. Specifies
            -- one password for all types of access to the file and its
            -- attributes
    actual-values            [1] IMPLICIT SET OF Access-Control-Element }
            -- The semantics of this attribute are described in ISO 8571-2
    Access-Control-Element              ::= SEQUENCE {
        action-list                     [0] IMPLICIT Access-Request,
        concurrency-access              [1] IMPLICIT Concurrency-Access OPTIONAL,
        identity                        [2] IMPLICIT User-Identity OPTIONAL,
        passwords                       [3] IMPLICIT Access-Passwords OPTIONAL,
        location                        [4] IMPLICIT Application-Entity-Title OPTIONAL }

    Access-Request                      ::= BIT STRING {
        read                            (0),
        insert                          (1),
        replace                         (2),
        extend                          (3),
        erase                           (4),
        read-attribute                  (5),
        change-attribute                (6),
        delete-file                     (7)  } (size (4...4))

    Concurrency-Access                  ::= SEQUENCE {
        read                            [0] IMPLICIT Concurrency-Key,
        insert                          [1] IMPLICIT Concurrency-Key,
        replace                         [2] IMPLICIT Concurrency-Key,
        extend                          [3] IMPLICIT Concurrency-Key,
        erase                           [4] IMPLICIT Concurrency-Key,
        read-attribute                  [5] IMPLICIT Concurrency-Key,
        change-attribute                [6] IMPLICIT Concurrency-Key,
        delete-file                     [7] IMPLICIT Concurrency-Key }

    Access-Passwords                    ::= SEQUENCE {
        read-password                   [0]  Password,
        insert-password                 [1] Password,
        replace-password                [2] Password,
        extend-password                 [3] Password,
        erase-password`                 [4] Password,
        read-attribute-password         [5] Password,
        change-attribute-password       [6] Password,
        delete-password                 [7] Password }

    Password                            ::= CHOICE {
                                        GraphicString,
                                        OCTET STRING }
-- Application-Entity-Title             -- The use of this attribute is for further study

    Concurrency-Key                     ::= BIT STRING {
        not-required                    (0),
        shared                          (1),
        exclusive                       (2),
        no-access                       (3) } (size (2...2))
```

ANNEX A

(to Recommendation T.434)

BFT Abstract Syntax Definition

```
    BFT-FORMAT { ccitt(0) recommendation(8) tseries(20) bft(434) version(0) }
    BEGIN
    -- EXPORTS Everything
    IMPORTS;

    BINARY-DATA-Message                 ::= [APPLICATION 23]
                                        IMPLICIT SEQUENCE OF SEQUENCE {
        protocol-version                [28] Protocol-Version DEFAULT {version-1},
        filename                        [0] IMPLICIT Filename-Attribute OPTIONAL,
        permitted-actions               [1] IMPLICIT Permitted-Actions-Attribute OPTIONAL,
        contents-type                   [2] Contents-Type-Attribute OPTIONAL,
        --  DEFAULT { UNSTRUCTURED BINARY }
        --  not specifying this attribute implies that data-file-content
        --  is unstructured binary

        storage-account                     [3] IMPLICIT GraphicString OPTIONAL,
        date-and-time-of-creation           [4] IMPLICIT GeneralizedTime OPTIONAL,
        date-and-time-of-last-modification  [5] IMPLICIT GeneralizedTime OPTIONAL,
        date-and-time-of-last-read-access   [6] IMPLICIT GeneralizedTime OPTIONAL,
        --  7 is reserved for date-and-time-of-last-attribute-modification
        identity-of-creator                 [8] IMPLICIT GraphicString OPTIONAL,
        identity-of-last-modifier           [9] IMPLICIT GraphicString OPTIONAL,
        identity-of-last-reader             [10] IMPLICIT GraphicString OPTIONAL,
        --  11 is reserved for identity-of-last-attribute-modifier
        --  12 is reserved for file-availability
        filesize                            [13] IMPLICIT INTEGER OPTIONAL,
        future-filesize                     [14] IMPLICIT INTEGER OPTIONAL,
        access-control                      [15] Access-Control-Attribute OPTIONAL,
        --  the use of this attribute is for further study
        legal-qualifications                [16] IMPLICIT GraphicString OPTIONAL,
        private-use                         [17] Private-Use-Attribute OPTIONAL,
        structure                           [18] IMPLICIT OBJECT IDENTIFIER OPTIONAL,
        application-reference               [19] IMPLICIT SEQUENCE OF GraphicString OPTIONAL,
        machine                             [20] IMPLICIT SEQUENCE OF GraphicString OPTIONAL,
        operating-system                    [21] IMPLICIT OBJECT IDENTIFIER OPTIONAL,
        recipient                           [22] IMPLICIT SEQUENCE OF GraphicString OPTIONAL,
        character-set                       [23] IMPLICIT OBJECT IDENTIFIER OPTIONAL,
        compression                         [24] IMPLICIT SEQUENCE OF GraphicString OPTIONAL,
        --  Indicates an optional compression applied to the content
        --  octets of the attribute data-file-content
        environment                         [25] IMPLICIT SEQUENCE OF GraphicString OPTIONAL,
        pathname                            [26] IMPLICIT SEQUENCE OF GraphicString OPTIONAL,
        user-visible-string                 [29] IMPLICIT SEQUENCE OF GraphicString OPTIONAL,
        data-file-content                   [30] EXTERNAL OPTIONAL } }

    Contents-Type-Attribute                       ::= {
    --  See annex B of ISO 8571-2 for more information
        document-type                       [0] IMPLICIT SEQUENCE {
                  document-type-name        [1] Document-Type-Name,
                  parameter                 [0] ANY OPTIONAL } }
        --  The actual types to be used for values of the parameter
        --  field are defined in the document-type-name.
        --  Currently, only UNSTRUCTURED TEXT and UNSTRUCTURED BINARY
        --  are supported.

    Document-Type-Name                            ::= OBJECT IDENTIFIER

    Entity-Reference                              ::= INTEGER {
        no-categorization-possible                                    (0),
        initiating-file-service-user                                  (1),
        initiating-file-protocol-machine                              (2),
        service-supporting-the-file-protocol-machine                  (3),
        responding-file-protocol-machine                              (4),
        responding-file-service-user                                  (5) }

    Filename-Attribute                            ::= SEQUENCE OF GraphicString

    Password                                      :: = CHOICE {
                                        GraphicString,
                                        OCTET STRING }

    Permitted-Actions-Attribute                   ::= BIT STRING {
    --  Actions available
        read                                (0),
        insert                              (1),
        replace                             (2),
        extend                              (3),
        erase                               (4) } (size (2...2))
```

```
Private-Use-Attribute           :: = SEQUENCE {
                                manufacturer-values       [0] EXTERNAL OPTIONAL
                                }

Protocol-Version                ::= IMPLICIT BIT STRING { version-1 (0) }

User-Identity                   ::= GraphicString

END
```

APPENDIX I

(to Recommendation T.434)

Use of ASN.1 for Encoding

I.1 *ASN.1 introduction*

The coding rules were developed using Abstract Syntax Notation (ASN.1) rules. These coding rules translate a binary file and its attributes into a binary range.

ASN.1 specifies a set of basic encoding rules that may be used to derive the specification of a transfer syntax for values of types defined, using the notation specified in ISO 8824. These basic encoding rules are also to be applied for decoding such a transfer syntax in order to identify the data values being transferred.

The coding rules for BFT appear in this Appendix. They describe a transfer syntax, using ASN.1 notation that is very similar to that used by FTAM.

Following is a brief overview of how values are coded using types defined in ASN.1. For a complete description, see CCITT Recommendations X.208 and X.209.

I.2 *Structure of an Encoding*

Using ASN.1, the encoding of a data value shall consist of four components which shall appear in the following order:

1) identifier octet;

2) length octets;

3) contents octets;

4) end-of-contents octets.

Figure I-1/T.434 illustrates the structure of an encoding. Note that the length of the contents octets can be specified by either the length octets or an end-of-contents octets.

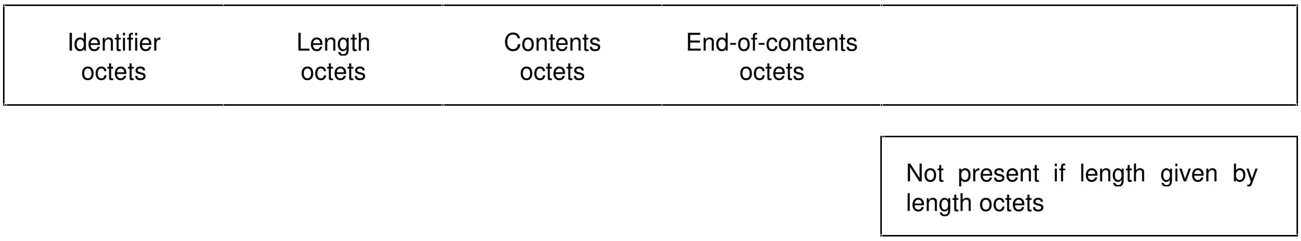

FIGURE I-1/T.434

The structure of a data value encoding

I.3 *Identifier octet*

The identifier octet encodes the ASN.1 tag (class and number) of the type of the data value. This tag is used to identify the data value in the context in which it is coded. The value of the tag can be determined explicitly or implicitly from the production rules being applied.

I.4 *Length octets*

The length octet encodes the length of the contents octets. Two forms of length octets are specified. These are:

a) The definite form. The octets consist of one or more octets, and represent the number of octets in the contents octets.

b) The indefinite form. The octet indicates that the contents octets are terminated by end-of-contents octets, and consist of a single octet with a value of 80 hex.

I.5 *Contents octets*

The contents octets consist of zero, one or more octets, and encode the data value as specified in ISO 8824 and ISO 8825.

I.6 *End-of-contents octets*

The end-of-contents octets shall be present if the length is encoded as a single octet with a value of 80 hex, otherwise they shall not be present.

The end-of-contents octets shall consist of two zero octets.

APPENDIX II

(to Recommendation T.434)

Differences between BFT Syntax and FTAM Syntax

II.1 The following list contains the parameters that are supplementary compared with the FTAM syntax:

- protocol-version;
- structure;
- application-reference;
- machine;
- operating-system;
- recipient;
- character-set;
- environment;
- pathname;
- user-visible-string;
- data-file-content.

II.2 For the following list of parameters, the ASN.1 application-wide tags contained in the FTAM syntax have been modified or removed:

- BINARY-DATA-Message;
- Document-Type-Name;
- Access-Request;
- Access-Passwords;
- Password;
- Application-Entity-Title.