CII Standard Series

CII Syntax Rule
Version 3.00

March 1998

Center for the Informatization of Industry (CII)

Table of Contents

Part 1: Components for Interchange Structure

Foreword

The CII Syntax Rules proposed by the Center for the Informatization of Industry of the Japan Information Processing Development Center in July 1992 has been widely adopted by various industries including manufacturing industries as the EDI standard. These syntax rules are now being used in Japan as actual EDI standards. In order to meet the demand for stabilization and wider adoption of the rules, it was decided to accept the standards as a JIS standard in 1998.

CII Syntax Rule version 3.00 consists of the following four parts.

Part-1: Components

Part-2: Structure of message group

Part-3: Structure of short form message

Part-4: Security functions

1. Scope

This standard applies to components of electronic data interchange between industries, between government agencies and industries, and within government agencies and industries.

2. Normative references

The following standards are referred to in these rules. The latest version of each standard is used.

JIS X 0201: Code for information interchange

JIS X 0208: Code of the Japanese graphic character set for information interchange

JIS X 0221: Universal multiple-octet coded character set (UCS)

3. Notation

3.1 Description of data

a) Data lengths are given in bits or bytes (1 byte = 8 bits).

b) Data values are described as a 1-byte character string or 2-byte kanji string or in numeric characters.

c) 1-byte character strings are graphic characters described in (JIS-X0201) or hexadecimal.

d) 2-byte character strings are kanji characters described in (JIS-X0208) or hexadecimal.

e) Numeric values are described in decimal/hexadecimal.

f) Decimal values are strings consisting of the numeric characters from 0 to 9, and are described as character strings.

g) Hexadecimal values are consist of numeric characters from 0 to 9 and an alphabetic character (A, B, C, D, E or F). One byte is written, for example, as X'43'.

h) Binary data are strings of bits, unless otherwise indicated.

i) Unsigned binary values are given in hexadecimal. If the length (number of bits) of an unsigned binary is not a multiple of 8, zeroes should be added to the left of the value.

3.2 Data element symbols

In order to identify each data element, a data element name and a symbol composed of numeric characters or a three-character string are assigned to it.

4. Definitions

4.1 Interchange structure

4.1.1 Operation message groups

One or more message groups consist of special messages.

4.1.2 Storage structure

Storage structure refers to the method used to store message groups in a file. When stored in a structured file, logical records are mapped to physical records. When stored in a non-structured file, the file is regarded as a structured record composed of 251-byte fixed length records, and logical records are mapped to physical records.

4.1.3 Storage mode

An identifier to show the method used to store message groups in a file.

4.1.4 Variable length physical record

A physical record which is variable in length.

4.1.5 Variable length dividing mode

Mode used to store message groups in a variable length record file.

4.1.6 Variable length record file

A file composed of one or more variable length physical records.

4.1.7 Structured file

A file composed of one or more physical records. Two file types are provided, a variable length record file and a fixed length record file.

4.1.8 Fixed length physical record

A physical record which is fixed in length.

4.1.9 Fixed length dividing mode

Mode used to store message groups in a fixed length record file or a non-structured file.

4.1.10 Fixed length record file

A file composed of one or more fixed length physical records.

4.1.11 Application message groups

One or more message groups composed of messages (of one or more types) and/or binary data.

4.1.12 Interchange unit

A transfer unit composed of one or more message groups on the telecommunications system. All message groups in one interchange unit are transferred from one and the same sending point to one and the same destination point.

4.1.13 Short form message group

A message group whose message group trailer is omitted under a special condition.

4.1.14 Broadcast message

One message group to which a broadcast header is added before the message group header. This message indicates that the message group with the broadcast header is transferred to the multiple destination points specified in the header.

4.1.15 Translator

The translator is an EDI tool that generates message groups according to the standard format from data groups in the EDI users system and carries out reverse processing.

4.1.16 Non-structured file

A file composed of one variable length record.

4.1.17 File

A file is a logical management unit of an auxiliary storage unit in a computer system. Two file types are provided, a structured file and a non-structured file.

4.1.18 Physical records

A physical record is an access unit divided optionally in a structured file. Two record types are provided, a fixed length record and a variable length record.

4.1.19 Message group

A message group is composed of messages (of one or more types) and/or binary data. A message group header is placed at the beginning of the message group and a message group trailer is placed at the end. A message group composed of only a message group header and a message group trailer with no message nor binary data also exists.

4.1.20 Logical record

The basic component of a message group. A logical record is a set of data elements and the byte string length is fixed (251 bytes) or variable. Twelve record types are provided as shown in Annex 1.

4.2 Logical record

4.2.1 Security trailer

A special logical record used in a pair with a security header to protect the whole message group.

4.2.2 Security header

A special logical record used to protect the whole message group. A security header cannot be used for a short form message group.

4.2.3 Error information messages

A special logical record for error information message groups. An error information message is a 251-byte fixed length logical record composed of fixed length data elements.

4.2.4 Transaction messages

A logical record for business data interchange. Usually, it is a variable length logical record composed variable length data elements. It is a component of application messages.

4.2.5 Receiving acknowledge message

A special logical record for receiving acknowledge message groups. A receiving acknowledge message is a 251-byte fixed length logical record composed of fixed length data elements.

4.2.6 Broadcast header

A special logical record placed before a message group header to indicate that the message group or short form message group is a broadcast message. A broadcast header is a 251-byte fixed length logical record composed of fixed length data elements.

4.2.7 Special message

A logical record in which information necessary to operate the EDI system is stored. This message is a 251-byte fixed length byte string. Two message types are provided, receiving acknowledge messages and error information messages.

4.2.8 Binary unit

A logical record in which the EDI user's bit string data is stored in bytes.

4.2.9 Binary data

A logical record group in which huge bit strings of the EDI user are stored. Binary data is formed of three types of logical records, a binary data header, a binary unit, and a binary data trailer.

4.2.10 Binary data trailer

A logical record that ends one binary data item. A binary data trailer is a 251-byte fixed length logical record composed of fixed length data elements.

4.2.11 Binary data header

A logical record placed at the beginning of one binary data for identification. A binary data header is a 251-byte fixed length logical record composed of fixed length data elements.

4.2.12 Sub security header message

A special logical record used to protect part of a message group or a short form message group.

4.2.13 Message

An identified, named and structured set of functionally related data elements, as described in a message specification. This set is stored in a logical record.

4.2.14 Message group trailer

A logical record placed at the end of one message group. A message group trailer is a 251-byte fixed length logical record composed of fixed length data elements.

4.2.15 Message group header

A logical record placed at the beginning of one message group for identification. A message group header is a 251-byte fixed length logical record composed of fixed length data elements.

4.3 Structure of logical record

4.3.1 Implicit repeating of user TFD set

A set of multiple user TFDs which have the same data tag number.

4.3.2 Variable length data element

A data element whose length is variable. Its maximum length is described in a data element specification. A data element name and/or a code name is used to identify a variable length data element. A data tag number is used as the code name and described in six decimal digits or five hexadecimal digits.

4.3.3 Repeat element

A set of repeat elements is a unit of multi detail and composed of single details and/or multi details.

4.3.4 Fixed length data element

A data element whose length is always fixed. A data element name or a code name is used to identify a fixed length data element. The code name is described as a character string containing one alphabetic character from A to Z and two numeric characters.

4.3.5 Control TFD

Placed in the TFD area to control the area.

4.3.6 Transfer form data element (TFD)

A form to handle variable length data elements. TFD data is composed of a data tag, a length tag and variable length data elements. Two TFD types are provided, control TFD and user TFD.

4.3.7 Transfer form data element area (TFD area)

A special area within a message where TFDs are placed. A message is composed of a message header and a TFD area.

4.3.8 Data tag

A binary number to identify the type and meaning of the variable length data element composing the TFD. A data tag is a bit string (1 to 3 bytes).

4.3.9 Length tag

A binary number that represents the length of the variable data element composing the TFD. A length tag is a 1-byte or 3-byte bit string.

4.3.10 Dividing identifier

A dividing identifier is a 1-byte fixed length data element. It is always placed at the left end of logical records and used to identify the logical record type. The logical record type is identified by the combination of a dividing identifier and a record identifier. Annex 1 shows the relationship between logical record types and dividing identifier values/record identifier values.

4.3.11 Multi detail

A multi detail is a set of repeat elements (each with a different value) which is a single repeat unit. There is a empty multi detail with no repeat elements. A multi detail is composed of a multi detail header, one or more repeat elements (or a blank if there is no repeat element), a return mark and a multi detail trailer.

4.3.12 Message header

A special area placed at the beginning of a message that identifies it uniquely. A message contains a message header and a TFD area.

4.3.13 User TFD

A TFD where an EDI user's data is stored within the TFD area.

4.3.14 Record identifier

Placed at the right of a dividing identifier in a logical record (except a binary unit). A record identifier is a 1-byte fixed length data element to identify the logical record type. The logical record type is identified by the combination of a dividing identifier and a record identifier. Annex 1 shows the relationship between logical record types and dividing identifier values/record identifier values.

4.4 Character set and character encoding rules

4.4.1 1-byte standard character set

Character set and encoding specified by JIS-X0201.

4.4.2 1-byte standard characters

1-byte characters included in the 1-byte standard character set.

4.4.3 2-byte standard character set

Character set and encoding specified by JIS-X0208 or JIS-X0221. It is prohibited to use both JIS-X0208 and JIS-X0221 within one message group.

4.4.4 2-byte standard characters

2-byte characters included in the 2-byte standard character set.

4.4.5 Limited standard character

Characters included in the limited standard character set. Message group header, message group trailer and broadcast header are formed of limited standard characters.

4.4.6 Limited standard character set

The limited standard character set consists of 38 characters including the numeric characters from 0 to 9 included in the 1-byte standard character set, the @ mark, the alphanumeric characters from A to Z, and a space character.

4.4.7 Standard numeric characters

Characters included in the standard numeric character set.

4.4.8 Standard numeric character set

The standard numeric character set consists of 1-byte numeric characters from 0 to 9. Encoding should be according to JIS-X0201.

4.4.9 Character set

The character set consists of characters which can be used by the EDI user. Two sets are provided, the 1-byte character set and the 2-byte character set.

5. Types of data elements and description of data types and data lengths

There are nine types of data elements according to data length (fixed or variable) and data type. The features and a description of each data element are given here. Annex 2 shows a data element in a standard message.

5.1 Types of fixed length data elements

a) Fixed length 1-byte data string

A fixed length byte string containing 1-byte data. The data type and length are described with "X[n]" (n: length in bytes). A fixed length 1-byte character string is also handled as a fixed length 1-byte data string.

b) Fixed length numeric data string

A fixed length (number of digits) byte string containing 1-byte numeric characters from 0 to 9. Data type and length are described with "9[n]" (n: number of digits).

c) Binary bit string

A fixed length byte string containing unsigned binary numbers. The data length is described in bytes. Binary bit string data elements are described as follows.

1-byte unsigned binary: "Bin8"

2-byte unsigned binary: "Bin16"

3-byte unsigned binary: "Bin24"

4-byte unsigned binary: "Bin32"

5.2 Types of variable length data elements

a) Variable length 1-byte data string

A variable length byte string containing 1-byte data (maximum length: 32767 bytes). The data type and length are described with "X(n)" (n: maximum length in bytes). A variable length 1-byte character string is also handled as a variable length 1-byte data string.

When a character string is included in a variable length 1-byte data string, encoding is done according to the data element, "Encoding mode for 1-byte characters," in the message group header.

If a character string formed of Shift JIS codes includes 2-byte character codes, the character string is handled as a variable 1-byte data string.

b) Variable length bit string in units of 1 byte

A variable length byte string composed of bit strings in units of 1 byte (1 byte = 8 bits, maximum length: 32767 bytes). The data type and length are described with "B(n)" (n: maximum length in bytes).

c) Variable length 2-byte data string

A variable length byte string containing 2-byte data (maximum length: 32767 bytes). The maximum number of 2-byte characters is 16,383. The data type and length are described with "K(n)" (n: maximum length in bytes). The maximum length should always be an even number.

When a 2-byte character string is included in a variable length 2-byte data string, encoding should be done according to the data element, "Encoding mode for 2-byte characters," in the message group header.

This byte string should be formed of only 2-byte characters. 1-byte characters cannot be included in this byte string.

d) Unsigned variable length numeric data string with implicit decimal point

A numeric data string whose number of digits is variable. The byte string is formed of 1-byte characters from 0 to 9 and the maximum length is thirty digits. While the number of integer part digits is variable, the number of fractional part digits is fixed. Therefore, a specified number of digits from the right of the numeric data string is regarded as the fractional part and the remaining digits to the left of the decimal part are regarded as the integer part. The data type, the maximum number of integer part digits, and the fixed number of fractional part digits are indicated as "9(n)V(m)" (n: maximum number of integer part digits, m: fixed number of fractional part digits). When the value of m is 0, the description of V(m) can be suppressed.

The characters forming this data string are encoded according to the data element, "Encoding mode for 1-byte characters," in the message group header.

e) Signed variable length numeric data string with explicit decimal point

A numeric data string whose number of digits is variable. This string contains 1-byte characters from 0 to 9, a positive/negative sign (+/-), and a decimal point (.) and the maximum length is thirty digits. The part to the left of the decimal point is regarded as the integer part and the part to the right of the decimal point is regarded as the fractional part. The data type and the maximum numbers of digits in the integer part and the fractional part are indicated as "N(n)V(m)" (n: maximum number of integer part digits, m: maximum number of fractional part digits). The sign and the decimal point are not counted in the maximum number of digits. When the value of m is 0, the description of V(m) can be suppressed.

Both a positive sign and a negative sign cannot be used in one numeric data string. A sign should be always placed at the beginning (left side) of a numeric data string. When the sign (+ or -) is suppressed, the numeric data string is assumed to be a positive number, and when the decimal point is suppressed, it is assumed to be an integer number.

The characters forming this numeric data string are encoded according to the data element, "Encoding mode for 1-byte characters," in the message group header.

f) Variable length date data

A variable length numeric data string to represent year, month and day (the Christian Era) in 1-byte characters. Two description types are provided, a 6-digit description (YYMMDD type) and an 8-digit description (YYYYMMDD type). The data type and number of digits are described with "Y(n)" (n: 6 or 8).

For the 6-digit description, the first two digits represent the year as follows.

1) 51 to 99: 1951 to 1999

2) 00 to 50: 2000 to 2050

6. Transfer form data elements

A transfer form data element (TFD) is a form to handle a variable length data element. A TFD contains a data tag, a data length tag, and a variable length data element, which are arranged consecutively from left to right. Annex 3 shows the structure of a TFD.

A TFD should be placed within the TFD area.

6.1 Structure of data tags

6.1.1 Length of data tag

A data tag contains bit string data with a length of 1 byte, 2 bytes or 3 bytes. The length is fixed according to the value of the first 1 byte (the leftmost 1 byte). Table 1 shows the relationship between the value of the first 1 byte of a data tag and the data tag length.

Table 1 Value of the first 1 byte of data tag and the data tag length

Value of the first 1 byte of data tag	Data tag length
X '00' to X 'EF'	2 bytes
X 'F0'	1 byte
X 'F1' to X 'F7'	3 bytes
X 'F8', X 'F9'	1 byte (for extension to features)
X 'FA'	2 bytes
X 'FB', X 'FC'	1 byte
X 'FD'	3 bytes
X 'FE'	1 byte
X 'FF'	1 byte (for extension to features)

A data tag with a first 1 byte whose value is X'F8', X'F9', or X'FF', has a length of 1 byte but the function is undefined and is available for feature extensions. At that time, the length may be changed.

6.1.2 Types of data tags

Two types are provided, a user data tag and a control data tag. The data tag type is fixed by the value of the first 1 byte in the data tag. Table 2 shows the relationship between the value of the first 1 byte in the data tag and the data tag type.

Table 2 Value of the first 1 byte in the data tag and the data tag type

Value of the first 1 byte in the data tag	Data tag type
X '00' to X 'EF'	User data tag
X 'F0'	Control data tag
X 'F1' to X 'F7'	User data tag
X 'F8' to X 'FF'	Control data tag

6.1.3 Function of user data tags

A user data tag identifies a variable length data element and its meaning. A user data tag value is called a data tag number and has the following meanings.

a) When the length of the user data tag is 2 bytes, the value is a 16-bit unsigned binary and represents a data tag number from 0 to 61439 (X'0000' to X'EFFF').

b) When the length of the user data tag is 3 bytes, the value is an unsigned binary (lower 19 bits) and represents a data tag number from 65536 to 524287 (X'F10000' to X'F7FFFF': upper five bits are ignored).

6.1.4 Function of control data tags

A control data tag controls the status within the TFD area in a message.

a) Value of the first 1 byte in the control data tag: X'F0'

The control data tag is placed at the beginning of the TFD area and works as a marker for the start of the TFD area.

b) Value of the first 1 byte in the control data tag: X'FA'

The control data tag is placed at the beginning of a multi detail and works as an A-type multi detail header. The length of an A-type multi detail header is 2 bytes and the value in the second byte indicates the detail number as a 1-byte unsigned binary. Values from X'31' to X'7E' can be used as detail numbers.

c) Value of the first 1 byte in the control data tag: X'FB'

The control data tag is placed between segments of a multi detail and works as a return mark to prompt for repeating a repeat element.

d) Value of the first 1 byte in the control data tag: X'FC'

The control data tag is placed at the end of a multi detail and works as a multi detail trailer.

e) Value of the first 1 byte in the control data tag: X'FD'

The control data tag is placed at the beginning of a multi detail and works as a D-type multi detail header. The length of a D-type multi detail header is 3 bytes and the values in the second and third bytes indicate the detail number as a 2-byte unsigned binary. Values from X'000A' to X'EFFF' can be used as detail numbers.

f) Value of the first 1 byte in the control data tag: X'FE'

The control data tag is placed at the end of the TFD area and works as a TFD trailer.

6.2 Structure of data length tags

A data length tag is used with a user data tag to specify the length of a variable length data element. A data length tag is described as a 1-byte or 2-byte unsigned binary number and represents the actual length of a variable length data element. The length of a data length tag is fixed to 1 byte or 3 bytes according to the value of the first 1 byte (the leftmost 1 byte) of the data length tag.

a) When the value of the first 1 byte of a data tag is any value from X'00' to X'FE', the length is 1 byte.

b) When the value of the first 1-byte of a data tag is X'F2', the length is 3 bytes.

A data length tag with a length of 1 byte is described as a 1-byte unsigned binary number and represents the actual length of the variable length data element immediately after the tag in bytes. The length ranges from 0 to 239 bytes (X'00' to X'EF').

A data length tag with a length of 3 bytes is described as a 2-byte unsigned binary number (the second and third bytes) and represents the actual length of the variable length data element immediately after the tag in bytes. The length ranges from 0 to 32767 bytes (X'0000' to X'7FFF').

In the case of a signed variable length numeric data string with explicit decimal point, the data length tag represents the length of the numeric data string including the sign (+/-) and the decimal point.

6.3 Structure of TFD

A TFD is a data string containing a data tag, a data length tag and a variable length data element, consecutively. The data length tag and/or the variable length data element may be suppressed. Two TFD types are provided, user TFDs and control TFDs.

a) A TFD which starts with a user data tag is called a user TFD. When the value of the data length tag is 0 (the length of a variable length data element = 0), the variable length data element is suppressed (the data length tag cannot be suppressed).

b) A TFD which starts with a control data tag is called a control TFD. A control data tag has no length tag nor variable length data element and is only used with a control data tag.

6.4 Compression and extension of length of variable length data elements

The length of a variable length data element in the TFD can be compressed or extended if the condition determined according to the data type is satisfied. The value of the data length tag in the TFD indicates the actual length of the variable length data element which was compressed or extended.

a) Variable length 1-byte data string

When the 1 byte placed at the rightmost of a data string is a space character, compressing the length by 1 byte by omitting the character will not change the value (or meaning) of the data element. This normative can also be applied to compressed data strings. When a variable length 1-byte data string contains only space characters, all characters are omitted and the data becomes a data string with a length of 0.

On the other hand, the length of a data string can be extended by adding a space character at the end of the data string.

b) Variable length bit string in units of 1 byte

When the 1 byte placed at the rightmost of a bit string in units of 1 byte is X'00', compressing the length by 1 byte by omitting the X'00' will not change the value (or meaning) of the data element. This normative can be also applied to compressed bit strings. When a variable length bit string in units of 1 byte contains only X'00', all bytes are omitted and the bit string becomes a bit string with a length of 0.

On the other hand, the length of the bit string can be extended by adding X'00's to the end of it.

c) Variable length 2-byte data string

When the 2 bytes placed at the rightmost of a data string is a space character, compression of the length by 2 bytes by omitting the character will not change the value (or meaning) of the data element. This normative can be also applied to compressed data strings. When a variable length 2-byte data string contains only space characters, all characters are omitted and the data string becomes a data string with a length of 0.

On the other hand, the data string can be extended in units of 2 bytes by adding a space character to the end of it.

d) Unsigned variable length numeric data string with implicit decimal point

When the leftmost digit of a numeric data string is 0, compression by omitting the digit will not change the value of the data element. This normative can be also applied to compressed numeric data strings. When all the digits of an unsigned variable length numeric data string with implicit decimal point contain 0's, they are omitted and the numeric data string becomes a numeric data string with a length of 0.

On the other hand, the number of digits can be increased by adding 0's to the beginning of the numeric data string.

e) Signed variable length numeric data string with explicit decimal point

When the leftmost digit of a numeric data string or the digit on the right of a sign (+/-) is 0, compression by omitting the digit will not change the value of the data element. This normative can be also applied to compressed numeric data strings.

On the other hand, the number of digits can be increased by adding 0's to the right of the sign (+/-) of a numeric data string or to the beginning of an unsigned numeric data string.

When a numeric data string includes a decimal point and the rightmost digit is 0, compression by omitting the digit will not change the value of the data element. This normative can be also applied to compressed numeric data strings.

On the other hand, the number of digits of a numeric data string which includes a decimal point can be increased by adding 0 to the end of the string.

When an unsigned variable length numeric data string with explicit decimal point contains only a sign (+/-), a decimal point and/or 0's, they are omitted and the numeric data string becomes a numeric data string with a length of 0.

f) Variable length date data

When the leftmost digit of a numeric data string is 0, compression by omitting the digit will not change the value of the data element. This normative can be also applied to compressed numeric data elements.

On the other hand, the number of digits can be increased by adding 0's to the beginning of a numeric data string.

7. Structure of transfer form data element area

7.1 Transfer form data element area

A transfer form data element area (TFD area) is an area used to store the user TFD and control TFD within a message. Annex 4 shows the structure of the TFD area.

7.1.1 Basic structure of TFD area

A TFD area is a variable length byte string which begins with a TFD area header, user TFDs and/or multi details consecutively according to the standard message, and terminates in a TFD area trailer. It is a dummy control TFD (inoperative TFD) that a TFD area header within a TFD area except the left most TFD area header and does not affect the state of the TFD area.

A TFD area which has no user TFD or multi detail is also permitted and the smallest TFD area contains a TFD area header and a TFD area trailer.

7.1.2 The restriction for standard message design

The standard message should be designed within the permitted TFD area structure. Because a TFD area is composed of user TFDs and multi details, the standard message must be designed by combining variable length data element and multi details. In designing the standard message, control TFDs cannot be used.

7.2 Multi detail

A multi detail envelops multiple pairs of repeat elements and return marks, with an A-type or D-type multi detail header and a multi detail trailer. A multi detail is used to construct a repeat structure which a repeat element is the unit of repeating.

7.2.1 Multi detail header

A multi detail header is a control TFD that starts and identifies a multi detail. Two types are provided, an A-type multi detail header and a D-type multi detail header, and both of them have the same function.

An A-type multi detail header has a 1-byte multi detail number and a D-type multi detail header has a 2-byte multi detail number. Multi details in one message are identified by the combination of a multi detail header type and a multi detail number. For example, although the multi detail number X'31' of an A-type multi detail header and the multi detail number X'0031' of a D-type multi detail header have the same numeric value (decimal 49), they are identified as different multi details.

7.2.2 Repeating element

A repeating element is a byte string containing one or more user TFDs and/or multi details consecutively. This is the unit of repeating.

7.2.3 Return mark

A return mark is a control TFD placed at the end of a repeating element and indicates explicitly that the repeating element is to be repeated. A return mark can be placed at the end of an empty repeating element whose TFDs and/or multi details are omitted. In this case, two or more return marks are placed consecutively.

7.2.4 Multi detail trailer

A multi detail trailer is a control TFD placed at the end of a repeating element (or a multi detail) and terminates the multi detail.

When a return mark is placed at the left of a multi detail trailer, it is possible to omit the return mark and shift the multi detail trailer to the left by one byte. This normative can also be applied to a new multi detail that has been generated by shifting the multi detail trailer. If all repeating elements in a multi detail are empty, the multi detail contains only a multi detail header and a multi detail trailer.

7.2.5 Structure of multi detail and hierarchy

A multi detail contains an A-type or D-type multi detail header, pairs of repeating elements and return marks (the return mark for the last repeating element is omitted), and a multi detail trailer.

Multi details can be structured in a hierarchy by including multi details in repeating elements. In this case, a lower-level multi detail must be wholly included in the upper-level multi detail. When there are multiple lower-level multi details, each of them must be independent.

A lower-level multi detail can include multiple even lower-level multi details. A hierarchy can have any number of levels.

7.3 Identification of user TFD in TFD area

Each user TFD in a TFD area is uniquely identified by the data tag number of a user TFD or by the combination of a detail number of a multi detail and the data tag number of a user TFD.

7.3.1 Scope of unique identification

The scope within which a user TFD can be uniquely identified by the data tag number of a user TFD. A TFD area contains multiple scopes of unique identification. A user TFD in a TFD area can be identified uniquely by combining the detail number of a multi detail and the data tag number of a user TFD.

7.3.2 Basic scope of unique identification

A scope within a TFD area. Multi details are not included in this scope. In this scope, a user TFD is uniquely identified by the data tag number of the user TFD.

7.3.3 Scope of unique identification in multi detail

A scope within a repeating element composing a multi detail. Lower-level multi details are not included in this scope. In this scope, a user TFD is uniquely identified by the combination of the detail number of the multi detail and the data tag number of the user TFD.

Because a multi detail includes multiple repeat elements, there actually exist multiple scopes with the same detail number. Therefore, it is necessary to implicitly identify each repeat element in the same order as the order in which the data elements are arranged. The order in which the data elements are arranged must be carefully managed. If the order is changed, the meaning may also be changed.

7.4 Implicit repeating of a single user TFD

A set of multiple user TFDs that have the same data tag number within a unique identification scope. If there exist multiple user TFDs each having the same data tag number within one unique identification scope, it is regarded as a number of single user TFDs each having different variable length data element values and placed repeatedly. This is implicit repeating of a single user TFD.

7.5 Data tag numbers of user TFDs and detail numbers of multi details within the TFD area

In order to identify user TFDs and multi details in a TFD area uniquely, data tag numbers of user TFDs and detail numbers of multi details must satisfy the following rules.

a) Each data tag number within one unique identification scope should be a unique number.

b) When multiple user TFDs each having the same data tag number are placed within one unique identification scope, they are regarded as user TFDs of the same type (implicit repeating of a single user TFD).

c) If data tag numbers of user TFDs in all unique identification scopes within a TFD area are unique numbers, each user TFD can be uniquely identified through the TFD area only by the data tag number.

d) When multiple multi details exist in a TFD area, a detail number of every multi detail should be a unique number within the scope that includes multi details with A-type multi detail headers or D-type multi detail headers.

e) User TFDs and multi details can be placed in an arbitrary order within one unique identification scope. However, multi details should never overlap each other.

8. Structure of message group header

A message group header contains multiple fixed length data elements arranged consecutively from left (beginning of transfer) to right. Data elements and meanings are described below in the order in which the data elements are arranged from left to right. Data element symbols are given in parentheses. Annex 5 shows the data element format.

8.1 Dividing identifier (C01)

A data element, which length is 1 byte fixed, that identifies a message group header or trailer. The value is set at X'30'.

8.2 Record identifier (C02)

A data element, which length is 1 byte fixed, that identifies a message group header when the dividing identifier value is set at X'30'. The value is set at X'43'.

8.3 Operation mode (C03)

A data element, which length is 1 byte fixed, that identifies the mode of operation as test mode or normal mode. A value of X'31' identifies the test mode, and a value of X'20' or X'30' identifies normal operation mode.

8.4 Sending EDI service provider (C04)

A data element, which length is 12 bytes fixed, that indicates the coded name of the EDI service provider relaying the message group on the sending side. Described in twelve limited standard characters.

8.5 Sending center code (C05)

A data element, which length is 12 bytes fixed, that indicates the coded name of the computer center sending the message group. Described in twelve limited standard characters.

8.6 Sender code (C06)

A data element, which length is 12 bytes fixed, that indicates the coded name of the company or division sending the message group. Described in twelve limited standard characters.

8.7 Receiving EDI service provider (C07)

A data element, which length is 12 bytes fixed, that indicates the coded name of the EDI service provider relaying the message group on the receiving side. Described in twelve limited standard characters.

8.8 Receiving center code (C08)

A data element, which length is 12 bytes fixed, that indicates the coded name of the computer center receiving the message group. Described in twelve limited standard characters.

8.9 Receiver code (C09)

A data element, which length is 12 bytes fixed, that indicates the coded name of the company or division receiving the message group. Described in twelve limited standard characters.

8.10 BPID agency (C10)

A data element, which length is 4 bytes fixed, that indicates the coded name of the agency managing the standard message. Described in four limited standard characters.

8.11 BPID sub agency (C11)

A data element, which length is 2 bytes fixed, that indicates the internal identification code of the agency managing the standard message. Described in two limited standard characters.

8.12 BPID version number (C12)

A data element, which length is 2 bytes fixed, that indicates the version number of the standard message. Described in two limited standard characters.

8.13 Reserved area 1 (F11)

An area, which length is 12 bytes fixed, reserved for future extensions. Filled with the value X'20'.

8.14 Information type code (C14)

A data element, which length is 4 bytes fixed, that indicates the type of the standard message. Described in four limited standard characters.

8.15 Reserved area 2 (C15)

An area, which length is 3 bytes fixed, reserved for future extensions. Filled with the value X'20' or X'30'.

8.16 Reserved area 3 (C16)

An area, which length is 3 bytes fixed, reserved for future extensions. Filled with the value X'20' or X'30'.

8.17 Format identifier (C17)

A data element, which length is 2 bytes fixed, for identification. The meanings are as follows.

a) X'3130' ....Message group in the dividing variable length mode

b) X'3131' ....Message group in the dividing fixed length mode

c) X'3230' ....Receiving acknowledge message or error message

8.18 Interchange reference number (C18)

A data element, which length is 10 bytes fixed, that stores the symbol identifying the message group. Described in ten limited standard characters. When this data element is not used, it is filled with the value X'20'.

8.19 Date and time of creation (C19)

A data element, which length is 12 bytes fixed, to indicate the date and time of creation of the message group. Described in the YYMMDDHHMMSS format with twelve standard numeric characters. The first two digits represent the year as follows.

51 to 99: 1951 to 1999

00 to 50: 2000 to 2050

8.20 Reserved area 4 (F12)

An area, which length is 12 bytes fixed, reserved for future extensions. Filled with the value X'20'.

8.21 Syntax rule ID version number (C21)

A data element, which length is 6 bytes fixed, that indicates the agency managing the syntax rules and the version number. Described in six limited standard characters.

8.22 Reserved area 5 (C22)

An area, which length is 1 byte fixed, reserved for future extensions. The value is always set to X'45'.

8.23 Storage mode (C23)

A data element, which length is 1 byte fixed, that indicates the method of storing the logical record in the physical record. The mode is indicated as follows.

a) X'20' or X'4D' .... dividing fixed length mode

b) X'53' .... dividing variable length mode

8.24 1-byte character set (C24)

A data element, which length is 1 byte fixed, that indicates the encoding type of the 1-byte character set. The coding type is indicated as follows.

a) X'20' or X'53' .... 1-byte standard character set

b) X'4D' .... shift JIS (2-byte characters are acceptable.)

c) X'50' .... other character codes

8.25 2-byte character set (C25)

A data element, which length is 1 byte fixed, that indicates the encoding type of the 2-byte character set. The coding type is indicated as follows.

a) X'20' or X'53' .... 2-byte standard character set (JIS-X0208)

b) X'55' .... 2-byte standard character set (JIS-X0221)

c) X'4D' .... shift JIS (half-width characters are prohibited.)

d) X'50' .... other character codes

8.26 Reserved area 6 (C26)

An area, which length is 1 byte fixed, reserved for future extensions. The value is set to X'20' or X'53'.

8.27 Reserved area 7 (C27)

An area, which length is 5 bytes fixed, reserved for future extensions. Filled with the value X'20' or X'30'.

8.28 Reserved area 8 (C28)

An area, which length is 5 bytes fixed, reserved for future extensions. Filled with the value X'20' or X'30'.

8.29 Short form message group identifier (C29)

A data element, which length is 1 byte fixed, that indicates whether the message group is the short-form type or not. When the message is the short-form type, the value must be set to X'49', and when it is not the short-form type, the value must be set to X'20' or X'53'.

8.30 Controlling agency code for the sending EDI service provider (C30)

A data element, which length is 3 bytes fixed, that indicates the agency controlling the code representing the name of the sending EDI service provider relaying the message group on the sending side. Described in alphanumeric code with three limited standard characters.

8.31 Controlling agency code for the sending center code (C31)

A data element, which length is 3 bytes fixed, that indicates the agency controlling the code representing the computer center sending the message group. Described in alphanumeric code with three limited standard characters.

8.32 Controlling agency code for the sender code (C13)

A data element, which length is 3 bytes fixed, that indicates the agency controlling the code representing the company or division sending the message group. Described in alphanumeric code with three limited standard characters.

8.33 Controlling agency code for the receiving EDI service provider (C33)

A data element, which length is 3 bytes fixed, that indicates the agency controlling the code representing the name of the EDI service provider relaying the message group on the receiving side. Described in alphanumeric code with three limited standard characters.

8.34 Controlling agency code for the receiving center code (C34)

A data element, which length is 3 bytes fixed, that indicates the agency controlling the computer center receiving the message group. Described in alphanumeric code with three limited standard characters.

8.35 Controlling agency code for the receiver code (C35)

A data element, which length is 3 bytes fixed, that indicates the agency controlling the code representing the company or division receiving the message group. Described in alphanumeric code with three limited standard characters.

8.36 Reserved area 9 (F29)

An area, which length is 70 bytes fixed, reserved for future extensions. Filled with the value X'20'.

9. Message structure

A message contains a message header and a TFD area. Data elements and meanings are described as below in the order in which data elements are arranged from left (beginning of transfer) to right. Data element symbols are given in parentheses. Annex 5 shows the data element format.

9.1 Basic structure of message

Two message types are provided, A-type messages and B-type messages. An A-type message contains an A-type message header and a TFD area, and a B-type message contains a B-type message header and a TFD area.

9.2 Structure of A-type message header

This type of message header is used for a message whose length is less than or equal 32,768 bytes.

9.2.1 Dividing identifier (C01)

A data element which length is 1 byte fixed that identifies the message. Although the value is usually set to X'39', the value may change according to the file storage mode. (See Section 8, Part 2.)

9.2.2 Record identifier (C02)

A data element, which length is 1 byte fixed, that identifies the message category. The message type is indicated as follows.

a) X'44' .... transaction message
b) X'53' .... security header message
c) X'47' .... sub security header message
d) X'56' .... security trailer message

9.2.3 Sequence number (D03)

A data element, which length is 5 bytes fixed, that indicates the sequence in which messages are placed in one message group. Described in five standard numeric characters. The first number 00001 is assigned to the message immediately after the message group header and the following messages are numbered in ascending order.

9.2.4 Message length (D04)

A data element, which length is 2 bytes fixed, that indicates the message length. Described as a 16-bit unsigned binary number representing the value (message length - 1). The value ranges from X'000A' (00010) to X'7FFF' (32767). The actual message length ranges from 11 bytes to 32,768 bytes.

9.3 Structure of B-type message header

A message header used for a transaction message whose length falls in the range from 32,769 bytes to 10,000,000 bytes. This message header can be also used for messages whose length are less than or equal 32,768.

9.3.1 Dividing identifier (C01)

See 9.2.1.

9.3.2 Record identifier (C02)

A data element, which length is 1 byte fixed, that identifies the message category. The value is set at X'44' and a B-type message header can be used only for a transaction message.

9.3.3 Sequence number (D03)

See 9.2.3.

9.3.4 Message length (D04)

A data element, which length is 2 bytes fixed, that indicates the message length. The value is set at X'8080' because an extended message length is used for a B-type message header.

9.3.5 Identifier (D05)

An area, which length is 1 byte fixed, reserved for future extensions. The value is set to X'F7'.

9.3.6 Extended message length (D05)

A data element, which length is 7 bytes fixed, that indicates the message length. Described as a decimal number containing seven standard numeric characters representing the value (message length - 1). The value ranges from 18 to 9999999. The actual maximum length is 10,000,000 bytes.

9.4 Structure of TFD area

See Section 7, Part 1.

10. Binary data structure

Binary data takes the form of a logical data string which stores EDI user data in bytes as large bit strings and is used to interchange large bit string data such as CAD/CAM data and graphic data. The structure of a logical record string and fixed length data elements within each logical record arranged from left (beginning of transfer) to right are described below. Data element symbols are given in parentheses. Annex 6 shows the data structure and data element format.

10.1 Basic structure of binary data

Binary data is composed of a binary data header, one or more binary units and a binary trailer.

A binary data header marks the beginning of the binary data and identifies specific binary data among multiple blocks of binary data.

One or more binary units store EDI user data as large bit strings.

A binary data trailer terminates the binary data and marks the end of it.

10.2 Structure of binary data header

A binary data header is a logical record, which length is 251 bytes fixed, that indicates the beginning of the binary data. It contains the following fixed length data elements.

10.2.1 Dividing identifier (C01)

A data element which length is 1 byte fixed. When the value is set at X'40', it identifies the binary data header or the binary data trailer.

10.2.2 Record identifier (C02)

A data element which length is 1 byte fixed. When the value of the dividing identifier is set at X'40', the record identifier identifies the binary data header. The value should be set to X'48'.

10.2.3 Sequence number (D03)

A data element, which length is 5 bytes fixed, whose meaning is the same as the sequence number in the message header. Binary data is regarded to be equivalent to a message and given a number described in five standard numeric characters. For the numbering method, see Section 9.2.3, Part 1.

10.2.4 Relating number (H04)

A data element, which length is 4 bytes fixed, that indicates the logical relationship between the binary data and the message within one message group. It indicates that the binary data and the message bearing the same number are related logically. Character encoding follows the specifications set for the 1-byte character set (C24) in the message group header of the message group which includes this header.

In a message, a relating number is stored in the user TFD whose data tag number is X'EF00' (61184). The value of a relating number is determined by the operation but should be unique within the normal range.

10.2.5 File identifier (H05)

A data element, which length is 80 bytes fixed, used by the EDI user to identify binary data. Character encoding follows the specification set for the 1-byte character set (C24) in the message group header of the message group which includes this header. The identifier is determined by the operation but should be unique within a certain range.

10.2.6 Format identifier (H06)

A data element, which length is 32 bytes fixed, used by the EDI user to identify the format of large bit strings stored as binary data. Character encoding follows the specifications set for the 1-byte character set (C24) in the message group header of the message group which includes this header. The identifier is determined by the operation but should be unique within a certain range.

10.2.7 Compression identifier (H07)

A data element, which length is 32 bytes fixed, used by the EDI user to identify the compression method used on the large bit strings stored as binary data. Character encoding follows the specification set for the 1-byte character set (C24) in the message group header of the message group which includes this header. The identifier is determined by the operation but should be unique within a certain range.

10.2.8 Reserved area (F31)

An area, which length is 96 bytes fixed, reserved for future extensions. Filled with the value X'20'.

10.3 Binary unit

A logical record that stores EDI user data in bytes as large bit strings. Structure of the binary unit is described below. Any bit string whose size is not a multiple of 8 (ex. 185 bytes) is not acceptable.

10.3.1 Structure of binary unit

A binary unit is formed of a dividing identifier (1-byte fixed length data element) and an area that stores EDI user data in bytes as large bit strings.

10.3.2 Binary unit length

The binary unit length is fixed according to the value of the storage mode (C23) of the message group header.

When the storage mode value is X'53' (dividing variable length mode), each binary unit excluding the binary unit immediately before the binary data trailer is a fixed length logical record with a length of 32001 bytes. The binary unit immediately before the binary data trailer is a fixed length logical record with a length of 32001 bytes or a variable length logical record whose maximum length is 32001 bytes.

When the dividing mode value is X'20' or X'4D' (dividing fixed length mode), each binary unit is a fixed length logical record with a length of 251 bytes.

10.3.3 Dividing identifier (C01) and its values

A data element, which length is 1 byte fixed, that identifies both the binary units in the message group and the sequence. The value ranges from X'41' to X'49', according to the following rules:

a) The value X'49' is always assigned to the dividing identifier of the binary unit immediately before the binary data trailer.

b) The value X'41' is assigned to the dividing identifier of the binary unit immediately after the binary data header and X'42' is assigned to the dividing identifier of the next binary unit. Values X'43', X'44', X'45', X'46', X'47' and X'48' are assigned one by one to dividing identifiers of the following binary units sequentially. The value X'41' is assigned to the dividing identifier again next to that with the value X'48' and the sequence of X'41' to X'48' is repeated.

c) When binary data is composed of one binary unit, the above normative a) is applied. The value X'49' is assigned to the dividing identifier of the binary unit immediately after the binary data header.

10.3.4 Bit string storage area

A fixed length area with a length of 32000 bytes or 250 bytes that stores EDI user data in bytes as large bit strings. In the dividing variable length mode, only the area of the binary unit immediately before the binary data trailer is a variable length area whose maximum length is 32000 bytes.

When the large bit string holding EDI user data is longer than the bit string storage area, the bit string data is divided into multiple binary units.

Conversion should be never done when storing the bit string data in the bit string storage area.

This syntax normative does not apply to the format of bit string data (i.e., CAD/CAM data and graphic data).

10.3.5 Last binary unit and remainder

When EDI user data held as large bit strings is divided and stored in the bit string storage area, a remainder is left in the bit string storage area of the last binary unit because the length of the area is fixed. Therefore, the bit string data is left-justified with a trailing margin in the area. The length of the bit string data stored in the left part (effective length) is stored in the data element to indicate the effective length of the last bit string storage area (T05) of the binary data trailer. Values for the margin left in the right part of the area are not specified especially, but the margin is usually filled with the value X'20'.

In the dividing variable length mode, the last binary unit can be a variable length logical record of a required length that does not leave a margin in the bit string storage area of the binary unit. The length of the large bit strings of EDI user data (effective length) should be stored in the data element to indicate the effective length of the last bit string storage area (T05) of the binary data trailer.

10.4 Binary data trailer

A logical record, which length is 251 bytes fixed, that indicates the end of the binary data. A binary data trailer contains the following fixed length data elements.

10.4.1 Dividing identifier (C01)

See 10.2.1.

10.4.2 Record identifier (C02)

A data element which length is 1 byte fixed. When the value of the dividing identifier is set at X'40', the record identifier identifies the binary data trailer. The value should be set to X'54'.

10.4.3 Sequence number (C02)

See 10.2.3.

10.4.4 Relating number (H04)

See 10.2.4.

10.4.5 Effective length of the last bit string storage area (T05)

A data element, which length is 4 bytes fixed, that indicates the effective data length in the last bit string storage area. The large bit strings used to hold EDI user data are divided and stored into multiple bit string storage areas of fixed length according to the storage structure. Therefore, effective data is stored in the left part of the last bit string storage area, leaving a margin in the right part of the area. The number of bytes of the effective data is represented as a 32-bit unsigned binary number. Value 1 corresponds to one byte, value 2 to two bytes, and n to n bytes.

10.4.6 Total number of logical records (T06)

A data element, which length is 4 bytes fixed, indicating the sum of the total number of binary units existing between the binary data header and the binary data trailer and the added 2. The total number of logical records is represented as a 32-bit unsigned binary number. The total number includes logical records in the binary data header and the binary data trailer.

10.4.7 Reserved area (F41)

An area which length is 232 bytes fixed reserved for future extensions. Filled with the value X'20'.

10.5 Data tag number for EDI for design graphics

Sixteen TFDs each having a different data tag number (X'EF00' (61184) to X'EF0F' (61199)) are reserved as special TFDs for EDI used for design graphics. These special TFDs are used as user TFDs for design graphics within the TFD area. Contents of eight TFDs are specified as shown in Table 3 and the remaining eight TFDs are reserved for future extensions.

Table 3 Data tag number for EDI for design graphics

	Tag number		TFD name	Data type (Maximum length)
	Hex.	Dec.		
(1)	X' EF00'	61184	Sequence number	9 (5)
(2)	X' EF01'	61185	File identifier	X (80)
(3)	X' EF02'	61186	Format identifier	X (32)
(4)	X' EF03'	61187	Compression identifier	X (32)
(5)	X' EF0C'	61196	File name memo	X (250)
(6)	X' EF0D'	61197	File name memo	K (250)
(7)	X' EF0E'	61198	Free message	X (250)
(8)	X' EF0F'	61199	Free message	K (250)

a) The sequence number has the same meaning as the sequence number of the binary data header.

b) The file identifier has the same meaning as the file identifier of the binary data header.

c) The format identifier has the same meaning as the format identifier of the binary data header.

d) The compression identifier has the same meaning as the compression identifier of the binary data header.

e) The file name memo stores free memo used by the EDI user. This data element does not exist in the binary data header related to it.

f) The free message stores EDI user messages. This data element does not exist in the binary data header related to it.

11. Structure of special messages and broadcast header

A special message is a logical record, which length is 251 bytes fixed, containing fixed length data elements and used for EDI system operation. Although a special message is a kind of message (transaction message) and has the same logical characteristics, the internal structure of a special message are different from messages (transaction messages).

A broadcast header is used to notify EDI services that a message group is a broadcast message and the broadcast destinations. A broadcast header is a logical record, which length is 251 bytes fixed, containing fixed length data elements.

Fixed length data elements and fixed length areas in special messages and broadcast headers are described below in the order in which they are arranged from left (beginning of transfer) to right. Coded data element names are given in parentheses. Annex 7 (special messages) and Annex 8 (broadcast header) show data element formats.

11.1 Types of special messages

Special messages are classified into the following two types.

a) Receive acknowledge message

Used to notify the sender that the receiver has received an interpretable transaction message.

b) Error information message

Used by the EDI service provider to notify the sender of the occurrence of an error during the interchange service.

11.2 Structure of receive acknowledge message

11.2.1 Dividing identifier (C01)

A data element, which length is 1 byte fixed, that identifies a message. This area should be set value X'39'.

11.2.2 Record identifier (C02)

A data element, which length is 1 byte fixed, that identifies a transaction message. This area should be set value X'44'.

11.2.3 Sequence number (D03)

A data element, which length is 5 bytes fixed, that indicates the sequence in which messages are arranged within one message group. Described in five standard numeric characters. The first number 00001 is assigned to the message immediately after the message group header and the following messages are numbered in ascending order.

11.2.4 Contents of the first half of the receive message group header (E51)

An area, which length is 129 bytes fixed, that stores 129 bytes from the dividing identifier (C01) to the creation time (C19) in the header of a successfully received message group.

11.2.5 Contents of the first half of receiving message group trailer (E52)

An area, which length is 37 bytes fixed, that stores 37 bytes from the dividing identifier (C01) to the reserved area 2 (E05) of the trailer of a successfully received message group.

11.2.6 Error flag area (E55 to E59)

An area, which length is 10 bytes fixed, that stores an error code that indicates the type of error detected while processing a message group with a receiving translator. This area can stores five fixed length 2-byte error codes described in standard numeric characters. Annex 7 gives details, including error codes.

11.2.7 Creation date and time (E60)

A data element, which length is 12 bytes fixed, to indicate the creation date and time of the receive acknowledge message. This data element is described in the YYMMDDHHMMSS format in twelve standard numeric characters. The first two digits represent the year as follows.

51 to 99: 1951 to 1999
00 to 50: 2000 to 2050

11.2.8 Reserved area 1 (F61)

An area, which length is 56 bytes fixed, reserved for future extensions. Filled with the value X'20'.

11.3 Structure of error messages

11.3.1 Dividing identifier (C01)

See 11.2.1.

11.3.2 Record identifier (C02)

See 11.2.2.

11.3.3 Sequence number (D03)

See 11.2.3.

11.3.4 Contents of the first half of the receiving message group header (E71)

An area, which length is 162 bytes fixed, that stores 162 bytes from the dividing identifier (C01) to the reserved area 8 (C28) of the header of the message group containing the error.

11.3.5 Contents of the first half of the receiving message group trailer (E72)

An area, which length is 37 bytes fixed, that stores 37 bytes from the dividing identifier (C01) to the reserved area 2 (E05) of the trailer of the message group containing the error.

11.3.6 Error flag area (E75 to E79)

An area, which length is 10 bytes fixed, that stores an error code indicating the cause of the error. This area can store five fixed length 2-byte error codes. Annex 7 gives details, including error codes.

11.3.7 Creation date and time (E80)

A data element, which length is 12 bytes fixed, that indicates the creation date and time for an error message. This data element is described in the YYMMDDHHMMSS format in twelve standard numeric characters. The first two digits represent the year as follows.

51 to 99: 1951 to 1999
00 to 50: 2000 to 2050

11.3.8 Reserved area 1 (F81)

An area, which length is 23 bytes fixed, reserved for future extensions. Filled with the value X'20'.

11.4 Structure of broadcast header

11.4.1 Dividing identifier (C01)

A data element, which length is 1 byte fixed, that identifies a broadcast header in combination with a record identifier. The value is set at X'30'.

11.4.2 Record identifier (C02)

A data element, which length is 1 byte fixed, that identifies a broadcast header in combination with a dividing identifier. The value is set at X'42'.

11.4.3 Operation mode (C03)

A data element, which length is 1 byte fixed, that identifies the operation mode: test mode or normal operation mode. When the value is set to X'31', this data element identifies test mode, and when the value is set to X'20' or X'30', it identifies normal operation mode.

11.4.4 Continuation identifier (B03)

A data element, which length is 1 byte fixed, that indicates whether another broadcast header exists or not. When the value is set to X'43', this data element indicates that another broadcast header exists, and when the value is set to X'45', it indicates that no broadcast header follows.

11.4.5 Number of broadcast destinations (B04)

A data element, which length is 1 byte fixed, that indicates the number of broadcast destinations within the header. The number is described in one standard numeric character.

11.4.6 Broadcast destinations 1 to 5 (B11 to E56)

Areas to indicate broadcast destinations. One broadcast header can indicate five destinations maximum. Limited standard characters must be used. Annex 8 gives details of the data element format.

11.4.7 Reserved area 1 (F23)

An area, which length is 21 bytes fixed, reserved for future extensions. Filled with the value X'20'.

12. Structure of message group trailer

A message group trailer contains multiple fixed length data elements arranged side by side from left (beginning of transfer) to right. Data elements and meanings are described below in the order in which they are arranged from left to right. Data element symbols are given in parentheses. Annex 5 shows the data element format.

12.1 Dividing identifier (C01)

A data element, which length is 1 byte fixed, that indicates a message group header or trailer. The value is set at X'30'.

12.2 Record identifier (C02)

A data element which length is 1 byte fixed . When the dividing identifier value is set to X'30' and the record identifier value is set to X'45', this data element identifies a message group trailer.

12.3 Last sequence number (E03)

A data element, which length is 5 bytes fixed, that indicates the sequence number of the message or binary data immediately before the message group trailer. The value represented in five standard numeric characters should be identical to the value of the sequence number in the message header of the message or the binary data trailer of the binary data immediately before the message group trailer.

12.4 Reserved area 1 (E04)

An area, which length is 15 bytes fixed, reserved for future extensions. Filled with the value X'20' or X'30'.

12.5 Reserved area 2 (E05)

An area, which length is 15 bytes fixed, reserved for future extensions. Filled with the value X'20' or X'30'.

12.6 Reserved area 3 (F51)

An area, which length is 213 bytes fixed, reserved for future extensions. Filled with the value X'20'.

Annex 1 (Normative):
Types and dividing/record identifier for logical records

Table 1-1 Types and dividing/record identifier for logical records

Type	Symbol	Dividing identifier value	Record identifier value	Logical record length	Notes
Broadcast header	BCH	X'30'	X'42'	251-byte fixed length	
Message group header	MGH	X'30'	X'43'	251-byte fixed length	
Transaction message	TRM	X'39' (Note 1)	X'44'	Variable length	
Receive acknowledge message	AKM	X'39'	X'44'	251-byte fixed length	Type of transaction message (Note 2)
Error message	ERM	X'39'	X'44'	251-byte fixed length	Same as above.
Security header message	SHM	X'39' (Note 1)	X'53'	Variable length	
Sub security header message	SSH	X'39' (Note 1)	X'47'	Variable length	
Security trailer message	STM	X'39' (Note 1)	X'56'	251-byte fixed length	
Binary data header	BDH	X'40'	X'48'	Variable length	
Binary unit	BU	X'41 to X'49'	—	251 or 32001 bytes	See Note 3.
Binary data trailer	BDT	X'40'	X'54'	251-byte fixed length	Fixed length format
Message group trailer	MGT	X'30'	X'45'	251-byte fixed length	Same as above.

Note 1) In the dividing fixed length mode, the value ranges from X'31' to X'39'.

Note 2) The transaction message type is determined by the information type in the header of the message group which includes the message or by the information type in the transaction message. When the transaction message type is specified by both, the information type shown by message group header is given priority.

Note 3) A binary unit has no record identifier. Although a binary unit is a fixed length logical record, the last binary unit can be handled as a variable length logical record with a maximum length of 32001 bytes in the dividing variable mode.

Annex 2 (Normative):
Description of data elements in standard messages

1. Description of data type and length

Table 2-1 Description of data type and length

Data type			Description	Example of description and data	Notes
Character data	Variable length 1-byte data string	X attribute	X(n) n: Maximum byte length	Ex.) When X(8), ABCDEFGH	Length in bytes
	Variable length bit string in units of 1 byte	B attribute	B(n) n: Maximum byte length	Ex.) When B(3), X'F256AB'	Length in bytes. X'00' is prohibited.
	Variable length 2-byte data string	K attribute	K(n) n: Maximum byte length	Ex.) When K(10), industry and information	Length in bytes. Twice of the number of kanji characters.
Numeric data	Unsigned variable length numeric data string with implicit decimal point	9 attribute	9(n), 9(n) V(m) n: Number of integer digits m: Number of decimal digits	Ex.) When 9(5), 23456 Ex.) When 9(3)V(2), 3456 (decimal point between 4 and 5)	9(5)V(0) is the same as 9(5).
	Signed variable length numeric data string with explicit decimal point	N attribute	N(n), N(n) V(m) n: Number of integer digits m: Number of decimal digits	Ex.) When N(5), -23456 Ex.) When N(4)V(2), -2345.6	N(5)V(0) is the same as N(5).
	Variable length data of Date	Y attribute	Y (n) n: 6 or 8	Ex.) When Y(6), 930331 in YYMMDD type Ex.) When Y(8), 19930331 in YYYYMMDD type	Julian calendar (Note)

Note) When n is 6, the year is represented as follows.

51 to 99: 1951 to 1999

00 to 50: 2000 to 2050

2. Description of data element in standard message (informative)

Data tag number	Data element name	Meaning of data element	Data type and length
000001	Data processing No.	Processing sequence number of receive data on the receiving side	9(5)

Annex 3 (Normative): Types and structures of data tags, length tags and transfer form data elements

1. Type and structure of data tag

Table 3-1 Type and structure of data tag

	Value of the first 1 byte of the data tag	Data tag length	Structure (hex.)	Tag name	Meanings
*1	X' 00' to X' EF'	2 bytes	×ˌ×│×ˌ× 00~ EF │ 00~ FF	2-byte user data tag	XXXX represents a data tag number from 0 to 61439 (X'0000' to X'EFFF'). (Numbers from 61440 to 65535 are prohibited.)
*2	X' F0'	1 byte	FˌO	Start of TFD area	Start of TFD area
User tag	X' F1' to X' F7'	3 bytes	Fˌ1│×ˌ×│×ˌ× ≀ Fˌ7│×ˌ×│×ˌ×	3-byte user data tag	The right-hand 19-bit number represents a data tag number from 65536 to 524287 (X'F10000' to X'F7FFFF').
Control tag	X' F8'	Undefined		Reserved	Reserved for future extensions
	X' F9'	Undefined		Reserved	
	X' FA'	2 bytes	FˌA│×ˌ×	A-type multi detail header (1-byte detail number)	XX represents a detail number from 49 to 126 (X'31' to X'7E').
	X' FB'	1 byte	FˌB	Return mark	Separation of repeat details
	X' FC'	1 byte	FˌC	Multi detail trailer	End of multi detail
	X' FD'	3 bytes	FˌD│×ˌ×│×ˌ×	D-type multi detail header (2-byte detail number)	XXXX represents a detail number from 10 to 61439 (X'000A' to X'EFFF').
	X' FE'	1 byte	FˌE	End of TFD area	End of TFD area
	X' FF'	Undefined		Reserved	Reserved for future extensions

Note) *1: User tag, *2: Control tag

2. Type and structure of length tag

Table 3-2 Type and structure of length tag

Values of the first 1 byte of the length tag	Length tag length	Structure (hex.)	Length in hex.	Length in dec.	Notes
X' 00' to X' EF'	1 byte	×ˌ×	X' 00' to X' EF'	000 to 239	8-bit binary (XX) representing the following variable length data element
X' F2'	3 bytes	Fˌ2│×ˌ×│×ˌ×	X' 0000' to X' 7FFF'	00000 to 32767'	16-bit binary (XXXX) representing the length of the following variable length data element

3. Type and structure of TFD

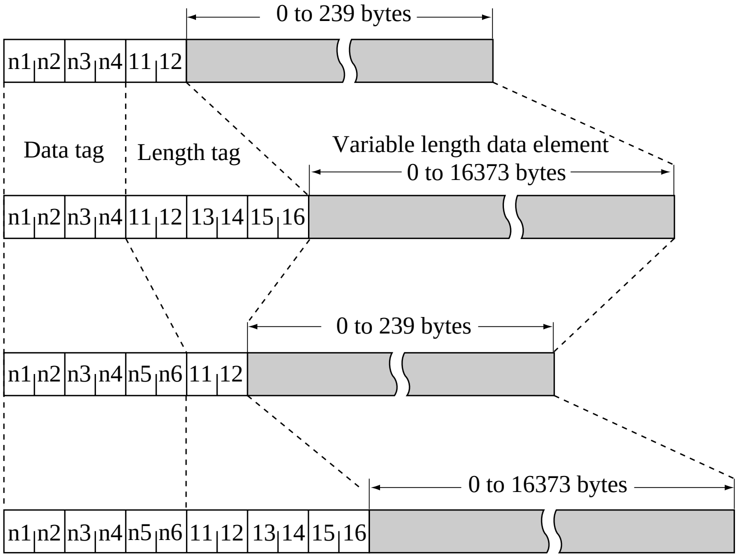
a) User TFD

1) Combination of 2-byte data tag and 1-byte length tag
   n1n2: X' 00'~X' EF, 1112: X' 00'~X' EF'

2) Combination of 2-byte data tag and 3-byte length tag
   n1n2: X' 00'~X' EF, 1112: X' F2'

3) Combination of 3-byte data tag and 1-byte length tag
   n1n2: X' F1'~X' F7, 1112: 'X 00'~X' EF'

4) Combination of 3-byte data tag and 3-byte length tag
   n1n2: X' F1'~X' F7, 1112: X' F2'



n3n4 : X' 00' ~ X' FF', n5n6 : X' 00' ~ X' FF'
13141516 : X' 0000' ~ X' 7FFF'

b) Control TFD

1) Start of TFD area
   (n1n2: X' F0')

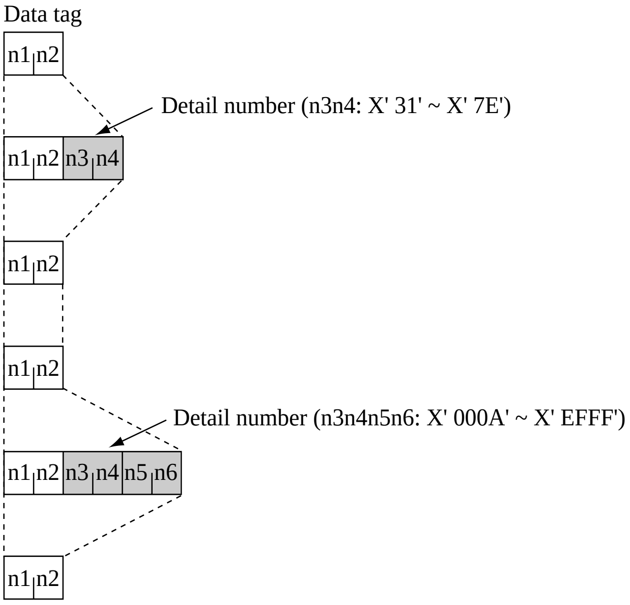
2) A-type multi detail header
   (n1n2: X' FA')

3) Return mark
   (n1n2: X' FB')

4) Multi detail trailer
   (n1n2: X' FC')

5) D-type multi detail header
   (n1n2: X' FD')

6) End of TFD area
   (n1n2: X' FE')


Data tag
n1 n2
Detail number (n3n4: X' 31' ~ X' 7E')
n1 n2 n3 n4
n1 n2
n1 n2
Detail number (n3n4n5n6: X' 000A' ~ X' EFFF')
n1 n2 n3 n4 n5 n6
n1 n2

Annex 4 (Normative): Structure of transfer form data element area and multi detail

1. Structure of transfer form data element (TFD) area

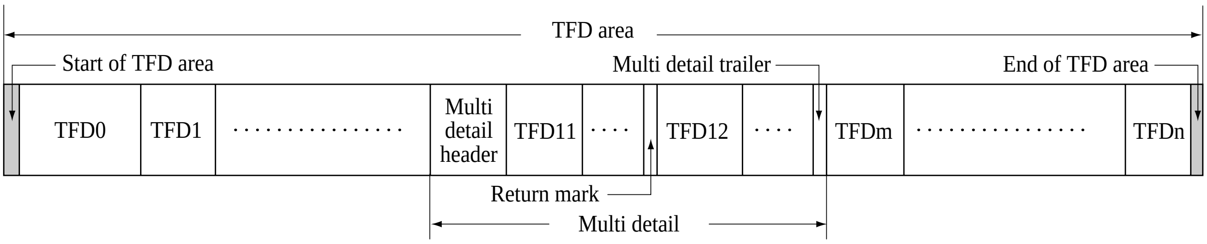


Fig. 4-1 Normal structure of TFD area

2. Structure of multi detail

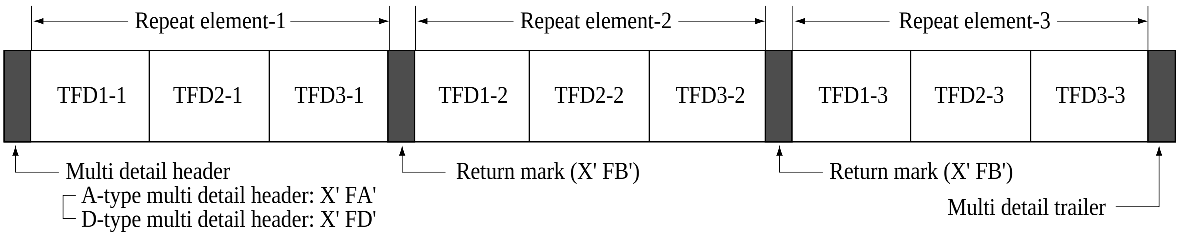
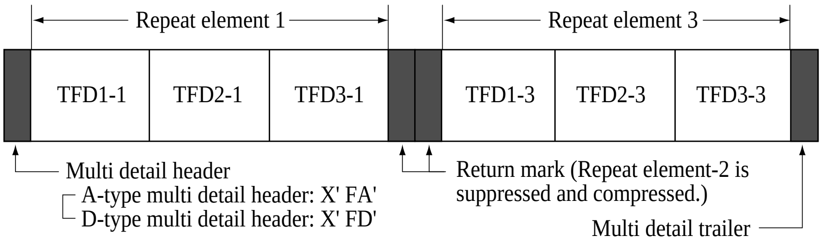


Fig. 4-2 Normal structure of multi detail



Repeat element 1
Repeat element 3
TFD1-1
TFD2-1
TFD3-1
TFD1-3
TFD2-3
TFD3-3
Multi detail header
A-type multi detail header: X' FA'
D-type multi detail header: X' FD'
Return mark (Repeat element-2 is suppressed and compressed.)
Multi detail trailer

Fig. 4-3 Structure when TFD1-2, TFD2-2 and TFD3-2 in Fig. 4-2 are suppressed and compressed

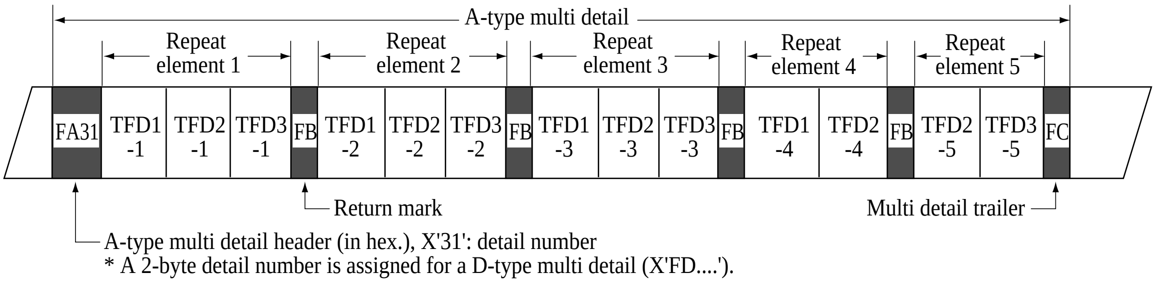


Fig. 4-4 Example of A-type multi detail

3. Meaning of multi detail

A multi detail corresponds to a REPEAT in structured COBOL.

		Structure		
		Structure component 1	Structure component 2	Structure component 3
Repeat element	1st.	TFD1-1	TFD2-1	TFD3-1
	2nd.	TFD1-2	TFD2-2	TFD3-2
	3rd.	TFD1-3	TFD2-3	TFD3-3
	4th.	TFD1-4	TFD2-4	— (Note)
	5th.	— (Note)	TFD2-5	TFD3-5

Note) Space character (for X type and K type), 0 (for 9 type and N type) or X'00'.

Fig. 4-5 Structure meant by the multi detail in Fig. 4-4

4. Multiplication of multi details

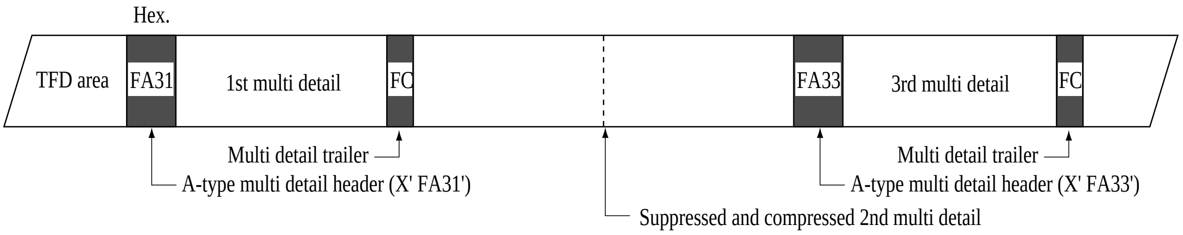
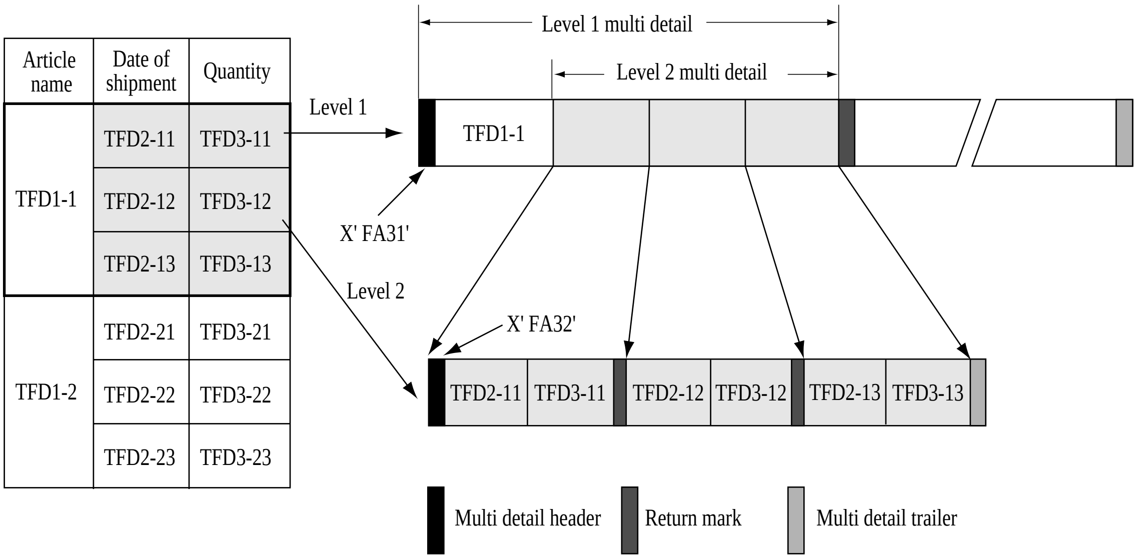

Hex.
TFD area
FA31
1st multi detail
FC
FA33
3rd multi detail
FC
Multi detail trailer
A-type multi detail header (X' FA31')
Multi detail trailer
A-type multi detail header (X' FA33')
Suppressed and compressed 2nd multi detail

Fig. 4-6 Example of multiplication of multi details
(The second multi detail is suppressed and compressed.)

5. Nesting multi details



Article name	Date of shipment	Quantity
TFD1-1	TFD2-11	TFD3-11
	TFD2-12	TFD3-12
	TFD2-13	TFD3-13
TFD1-2	TFD2-21	TFD3-21
	TFD2-22	TFD3-22
	TFD2-23	TFD3-23

Fig. 4-7 Example of a multi detail which includes a lower-level multi detail

6. Unique identification of user TFDs in multi details

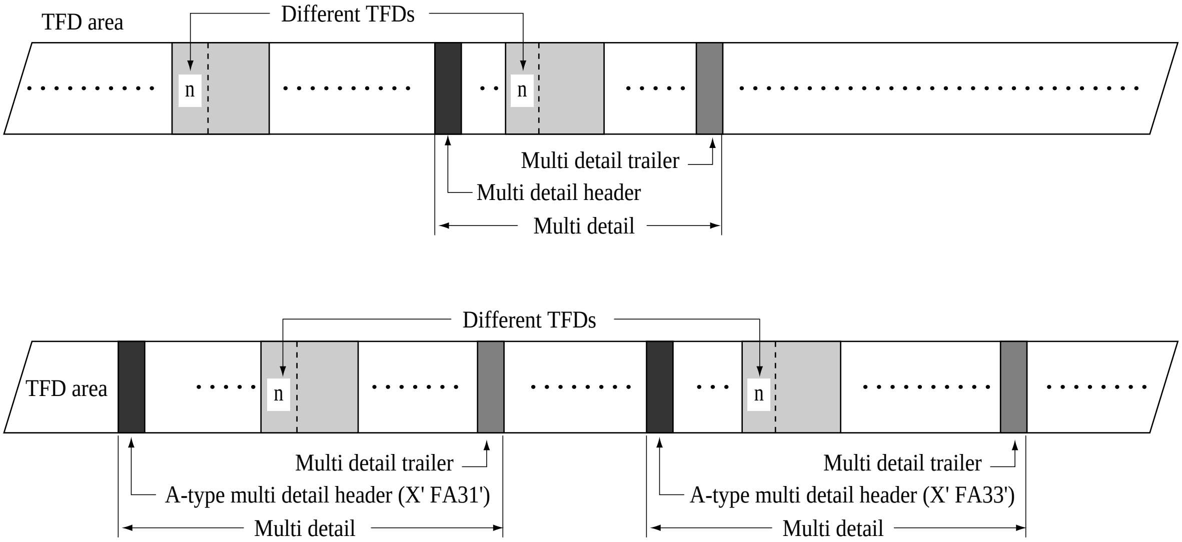
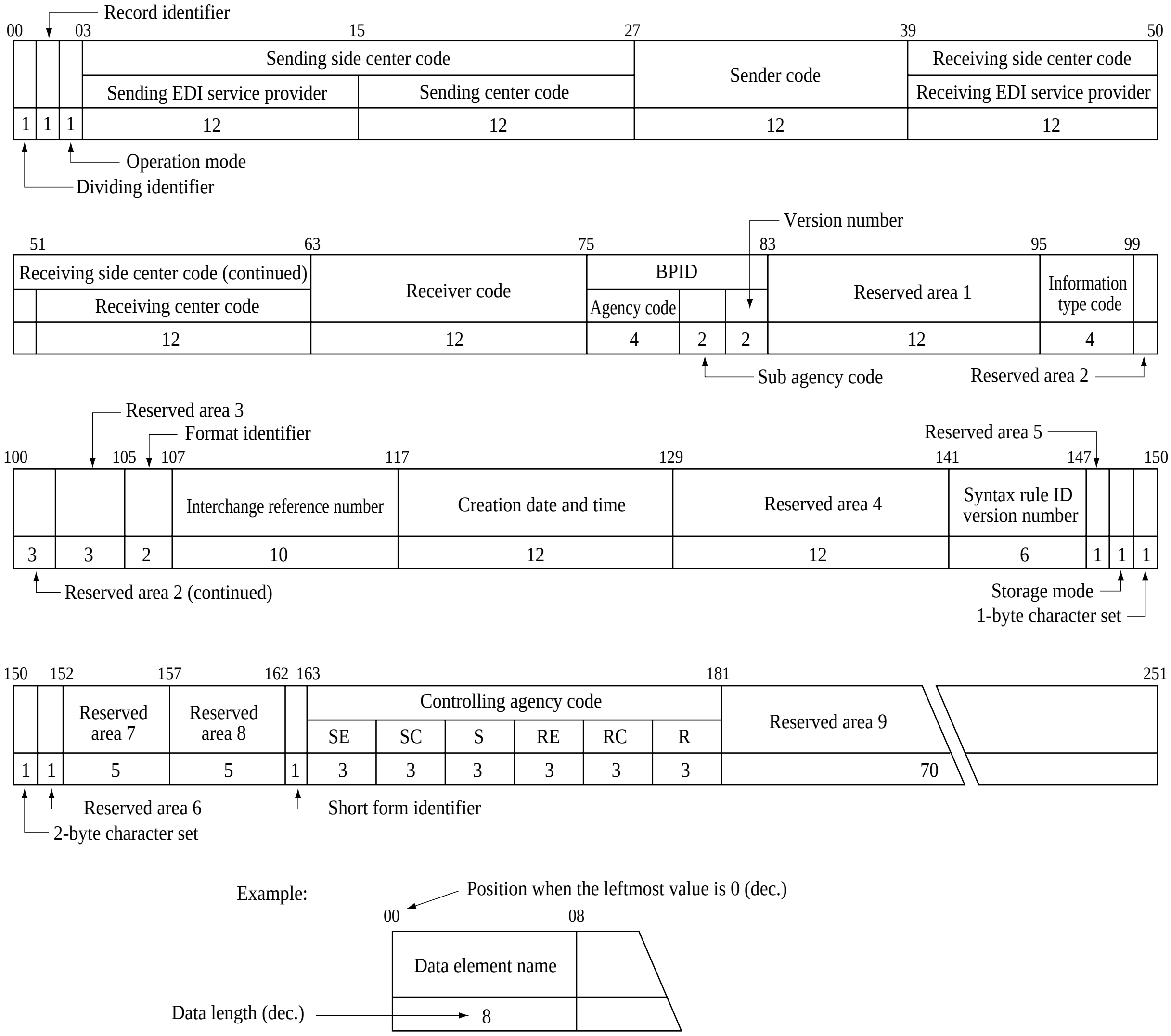


Fig. 4-8 Example of different TFDs with the same data tag number (n)

Annex 5 (Normative):
Data elements and format of basic logical records

1. Data elements and format of message group header



Note) SE: Controlling agency code for sending EDI service provider
SC: Controlling agency code for sending center code
S: Controlling agency code for sender code
RE: Controlling agency code for receiving EDI service provider
RC: Controlling agency code for receiving center code
R: Controlling agency code for receiver code

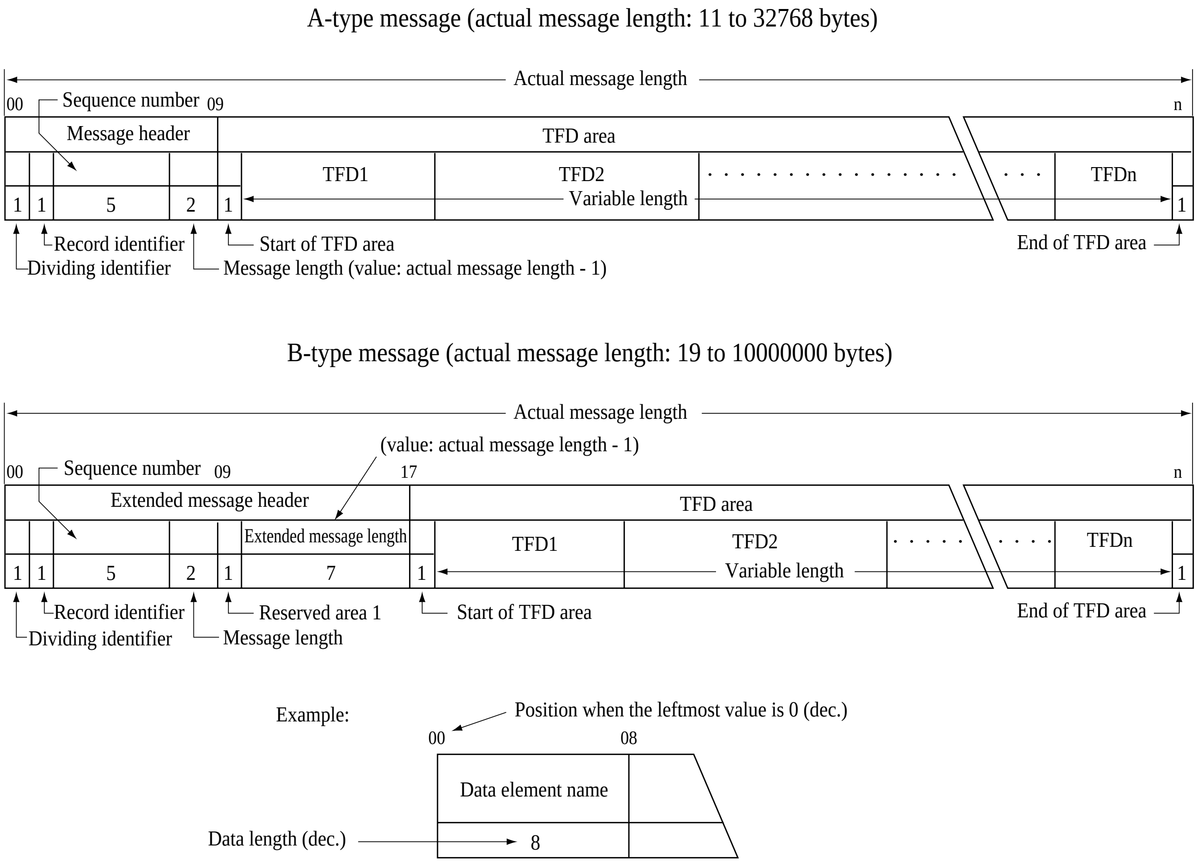
Fig. 5-1 Data element format of message group header

Table 5-1   Data elements of message group header

Symbol	Data type (length)	Data element	Notes (value to set)
C01	X[1]	Dividing identifier	Set to X'30'.
C02	X[1]	Record identifier	Set to X'43'.
C03	X[1]	Operation mode	X'20' or X'30': normal message, X'31': test message
C04	X[12]	Sending EDI service provider	Code described in twelve limited standard characters
C05	X[12]	Sending center code	Code described in twelve limited standard characters
C06	X[12]	Sender code	Code described in twelve limited standard characters
C07	X[12]	Receiving EDI service provider	Code described in twelve limited standard characters
C08	X[12]	Receiving center code	Code described in twelve limited standard characters
C09	X[12]	Receiver code	Code described in twelve limited standard characters
C10	X[4]	BPID agency code	Code described in four limited standard characters
C11	X[2]	BPID sub agency code	Code described in two limited standard characters
C12	X[2]	BPID version number	Code described in two limited standard characters
F11	X[12]	Reserved area 1	Filled with X'20'.
C14	X[4]	Information type code	Code described in four limited standard characters
C15	9[3]	Reserved area 2	Filled with X'20' or X'30'.
C16	9[3]	Reserved area 3	Filled with X'20' or X'30'.
C17	X[2]	Format identifier	X'3130': dividing variable length message, X'3131': dividing fixed length message, X'3230': receive acknowledge message or an error message
C18	X[10]	Interchange reference number	Identifier described in ten limited standard characters
C19	X[12]	Creation date and time	Described in the YYMMDDHHMMSS format in standard numeric characters.
F12	X[12]	Reserved area 4	Filled with X'20'.
C21	X[6]	Syntax normative ID version number	Code described in six limited standard characters that indicates the agency controlling the syntax normative and version number.
C22	X[1]	Reserved area 5	Set to X'45'.
C23	X[1]	Storage mode	X'53': dividing variable length mode, X'20' or X'4D': dividing fixed length mode
C24	X[1]	1-byte character set	X'20' or X'53': 1-byte standard character set, X'4D': shift JIS, X'50': other character set
C25	X[1]	2-byte character set	X'20' or X'53': JIS-X0208, X'55': JIS-X0221, X'4D': shift JIS (except half-width characters), X'50': other kanji character set
C26	X[1]	Reserved area 6	Set to X'20' or X'53'.
C27	9[5]	Reserved area 7	Filled with X'20' or X'30'.
C28	9[5]	Reserved area 8	Filled with X'20' or X'30'.
C29	X[1]	Short form message group identifier	X'20' or X'53': not short-form type, X'49': short-form type
C30	X[3]	SE	Described in three limited standard characters to indicate the agency controlling the code.
C31	X[3]	SC	Described in three limited standard characters to indicate the agency controlling the code.
C32	X[3]	S	Described in three limited standard characters to indicate the agency controlling the code.
C33	X[3]	RE	Described in three limited standard characters to indicate the agency controlling the code.
C34	X[3]	RC	Described in three limited standard characters to indicate the agency controlling the code.
C35	X[3]	R	Described in three limited standard characters to indicate the agency controlling the code.
F13	X[70]	Reserved area 9	Filled with X'20'.

Note) SE: Controlling agency code for the sending EDI service provider
SC: Controlling agency code for the sending center code
S: Controlling agency code for the sender code
RE: Controlling agency code for the receiving EDI service provider
RC: Controlling agency code for the receiving center code
R: Controlling agency code for the receiver code

2. Data elements and formats of A-type and B-type messages


A-type message (actual message length: 11 to 32768 bytes)

B-type message (actual message length: 19 to 10000000 bytes)

Example:

Fig. 5-2 Data element formats of A-type and B-type messages

Table 5-2 Data elements of A-type message header

Symbol	Data type (length)	Data element	Notes (value to set)
C01	X[1]	Dividing identifier	Set to X'39'.
C02	X[1]	Record identifier	X'44': transaction message, X'53': security header message, X'47': sub security header message, X'56': security trailer
D03	9[5]	Sequence number	Indicates the sequence in which messages are arranged in one message group. The first sequence number 1 is assigned to the message immediately after the message group header and the following messages are numbered in ascending order. Described in five standard numeric characters.
D04	Bin16	Message length	Described as a binary number representing the value "actual message length-1". Values from 10 to 32767 are permitted. When the actual message length is above 32768 bytes, a B-type message must be used.

Table 5-3 Data elements of extended message header for B-type messages

Symbol	Data type (length)	Data element	Notes (value to set)
C01	X[1]	Dividing identifier	Set to X'39'.
C02	X[1]	Record identifier	Set to X'44'. Identifies a transaction message.
D03	9[5]	Sequence number	Indicates the sequence in which messages are arranged in one message group. The first sequence number 1 is assigned to the message immediately after the message group header and the following messages are numbered in ascending order. Described in five standard numeric characters.
D04	Bin16	Message length	Set to X'8080'. Indicates that extended message length is used.
D05	X[1]	Reserved area 1	Set to X'F7'.
D06	9[7]	Extended message length	Described as a decimal number in seven standard numeric characters representing the value "actual message length-1". Values from 18 to 9999999 are permitted.

3. Data elements and format of message group trailer

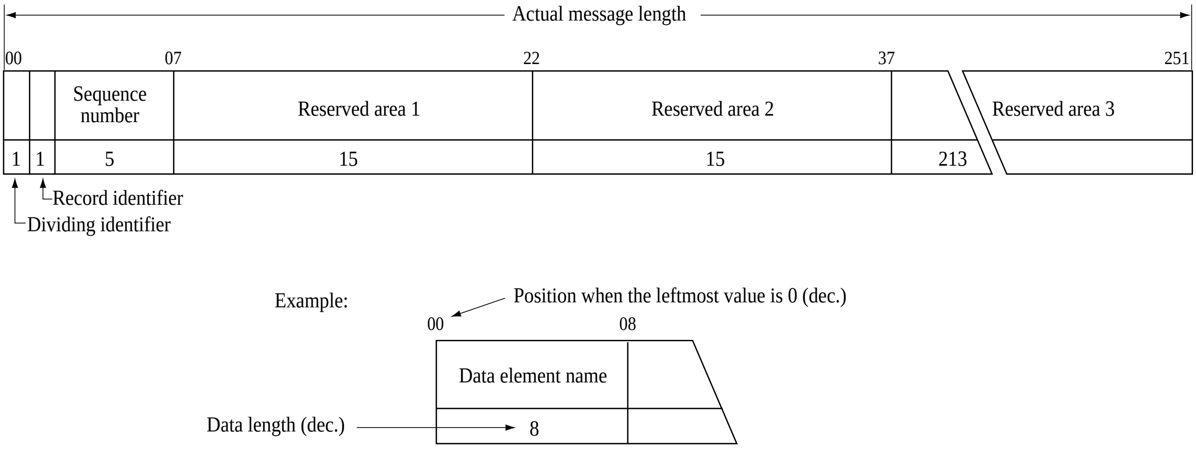


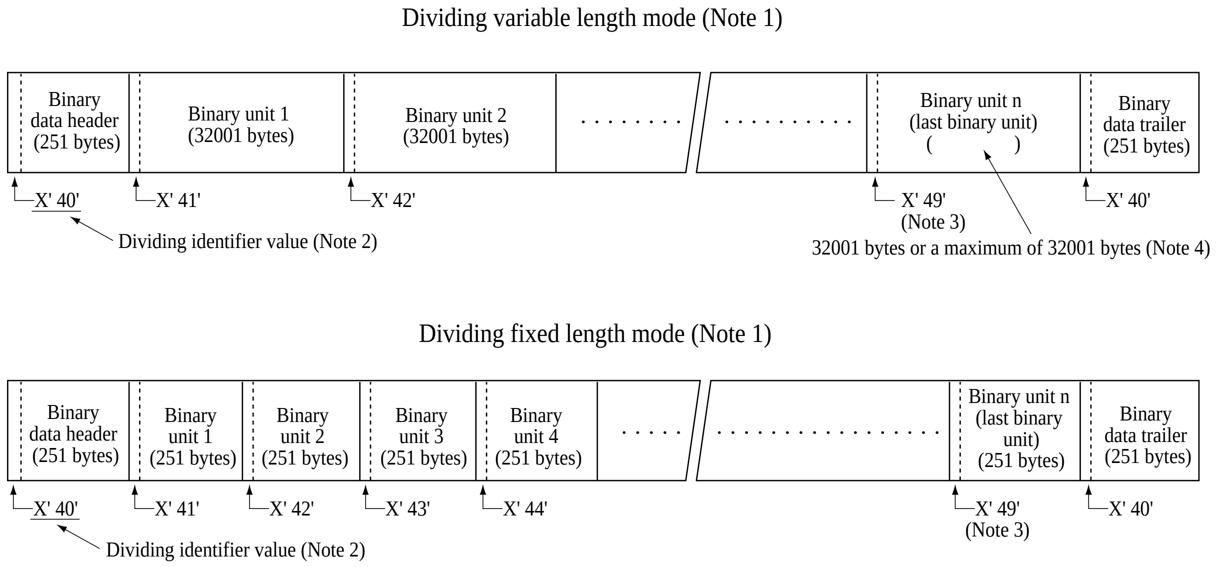
Fig. 5-3 Data element format of message group trailer

Table 5-4 Data elements of message group trailer

Symbol	Data type (length)	Data element	Notes (value to set)
C01	X[1]	Dividing identifier	Set to X'30'.
C02	X[1]	Record identifier	Set to X'45'.
E03	9[5]	Last sequence number	Set to the same value as that of the sequence number of the message immediately before the message group trailer. Described in five standard numeric characters.
E04	9[15]	Reserved area 1	Filled with X'20' or X'30'.
E05	9[15]	Reserved area 2	Filled with X'20' or X'30'.
F51	X[213]	Reserved area 3	Filled with X'20'.

Annex 6 (Normative):
Structure, data elements and format of binary data

1. Structure of binary data


Dividing variable length mode (Note 1)
Binary data header (251 bytes)
Binary unit 1 (32001 bytes)
Binary unit 2 (32001 bytes)
. . . .
. . . .
Binary unit n (last binary unit) (　　)
Binary data trailer (251 bytes)
X' 40'
X' 41'
X' 42'
X' 49' (Note 3)
X' 40'
Dividing identifier value (Note 2)
32001 bytes or a maximum of 32001 bytes (Note 4)
Dividing fixed length mode (Note 1)
Binary data header (251 bytes)
Binary unit 1 (251 bytes)
Binary unit 2 (251 bytes)
Binary unit 3 (251 bytes)
Binary unit 4 (251 bytes)
. . . . .
. . . .
Binary unit n (last binary unit) (251 bytes)
Binary data trailer (251 bytes)
X' 40'
X' 41'
X' 42'
X' 43'
X' 44'
X' 49' (Note 3)
X' 40'
Dividing identifier value (Note 2)

Note 1) The mode (dividing variable length mode or dividing fixed length mode) is determined by the storage mode value in the header of the message group in which the binary data is included.

Note 2) Dividing identifier values of a binary data header and a trailer are set to X'40'. Values X'41' to X'48' are assigned to binary unit dividing identifiers repeatedly (the value next to X'48' is X'41').

Note 3) The dividing identifier value of the last binary unit is X'49'.

Note 4) A fixed length logical record with a length of 32001 bytes or a variable length logical record whose maximum length is 32001 bytes can be selected as the last binary unit in the dividing variable mode.

Fig. 6-1 Structure of binary data

2. Data elements and format of binary data header

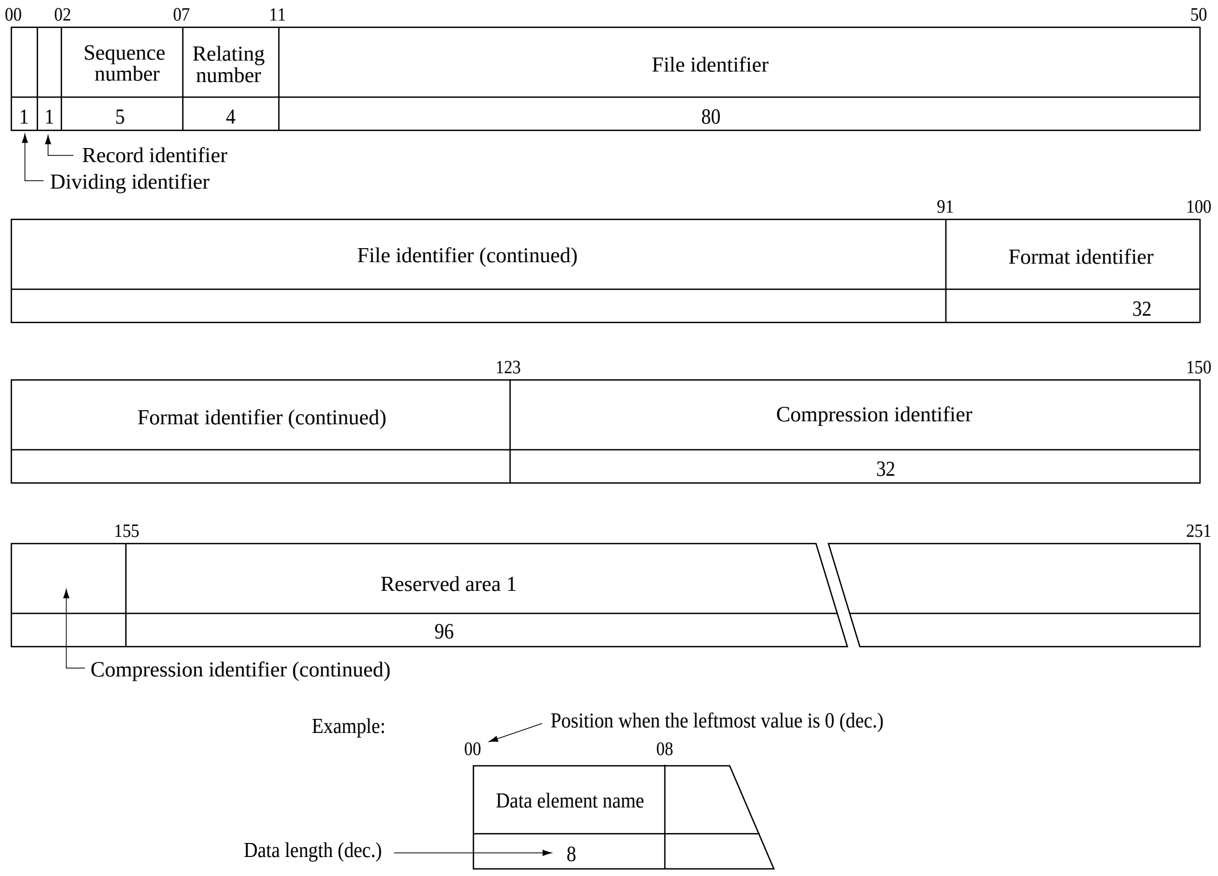
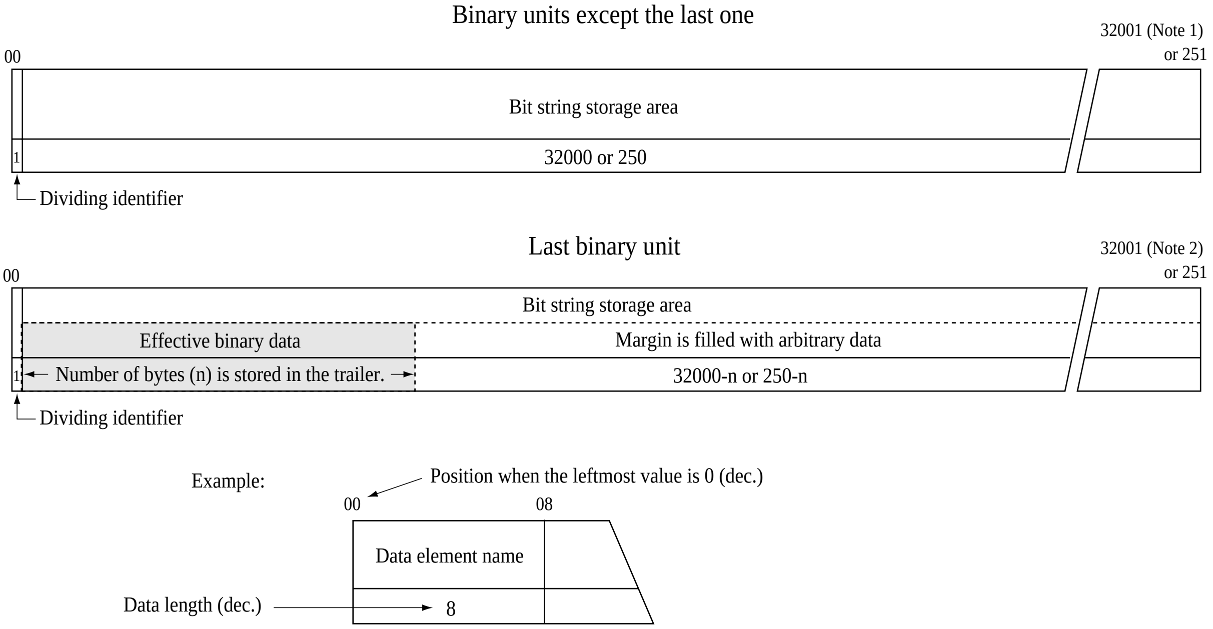


Fig. 6-2 Data element format of binary data header

Table 6-1 Data elements of binary data header

Symbol	Data type (length)	Data element	Notes (value to set)
C01	X[1]	Dividing identifier	Identifies binary data header or binary data trailer. Set to X'40'.
C02	X[1]	Record identifier	Identifies binary data header when the dividing identifier value is set to X'40'. Set to X'48'.
D03	9[5]	Sequence number	A number described in five standard numeric characters having the same meaning as the sequence number of the message header. For the numbering method, see Section 9.2.3, Part 1.
H04	9[4]	Relating number	Indicates the logical relationship between the message and binary data within one message group.
H05	X[80]	File identifier	Identifies binary data used by the EDI user. When the identifier length falls short of 80 bytes, the identifier should be left-justified and the margin is filled with space characters.
H06	X[32]	Format identifier	Identifies format of binary data used by the EDI user. When the identifier length falls short of 32 bytes, the identifier should be left-justified and the margin is filled with space characters.
H07	X[32]	Compression identifier	Identifies the compression method used for binary data used by the EDI user. When the identifier length falls short of 32 bytes, the identifier should be left-justified and the margin is filled with space characters.
F31	X[96]	Reserved area 1	Filled with X'20'.

3. Data elements and format of binary unit



Note 1) In the dividing variable length mode, the binary unit length is 32001 bytes, and in the dividing fixed length mode, it is 251 bytes.

Note 2) In the dividing variable length mode, the last binary unit can be a variable length logical record whose maximum length is 32001 bytes.

Fig. 6-3 Data element format of binary unit

Table 6-2 Data elements of binary unit

Symbol	Data type (length)	Data element	Notes (value to set)
C01	X[1]	Dividing identifier	Indicates a binary unit. The value X'41' is assigned to the dividing identifier of the binary unit immediately after the binary data header. Values X'41' to X'48' are repeatedly assigned to dividing identifiers of the following binary units except for the last one. The value assigned to the dividing identifier of the last binary unit is X'49'.
D10	X(32000) or X[250]	Bit string storage area	Area that stores EDI user data as large bit strings. The length of this area is fixed according to the value of the storage mode (C23) of the message group header (the dividing variable length mode: 32000 bytes, the dividing fixed length mode: 250 bytes). Regarding the last binary unit, the length of this area can be variable (maximum length: 32000 bytes) only in the dividing variable length mode. In the last binary unit, large bit strings holding EDI user data are stored left-justified and the length (effective length) is stored in the effective length of the last bit string storage area in the binary data trailer.

4. Data elements and format of binary data trailer

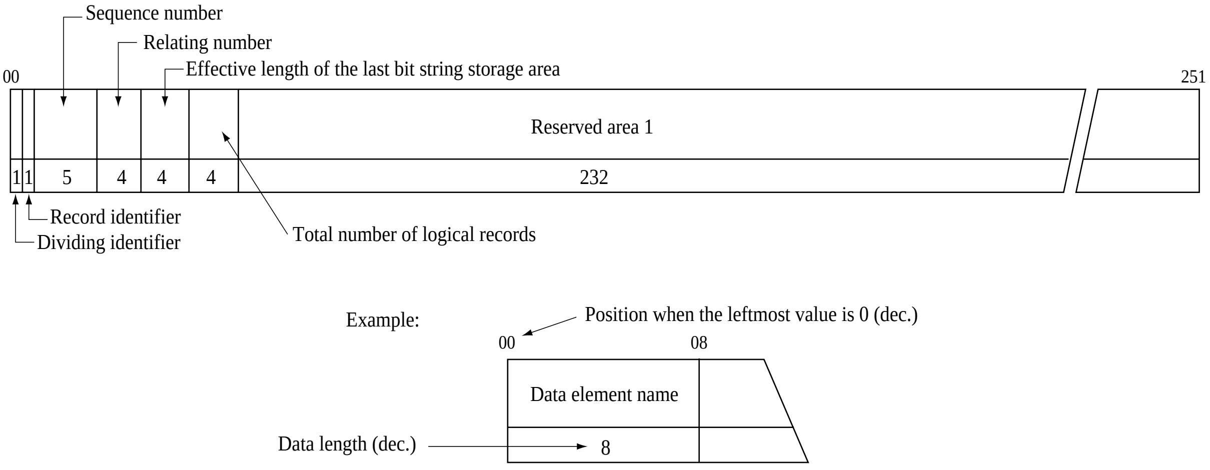
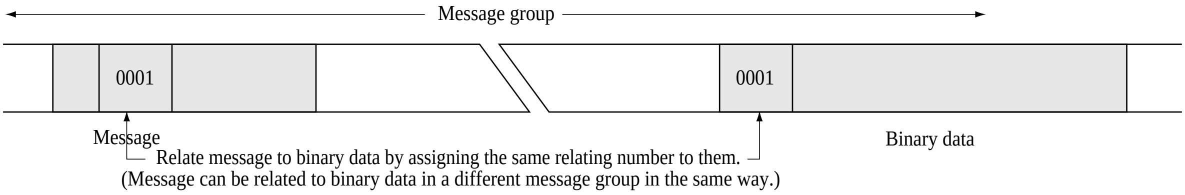


Fig. 6-4 Data element format of binary data trailer

Table 6-3 Data elements of binary data trailer

Symbol	Data type (length)	Data element	Notes (value to set)
C01	X[1]	Dividing identifier	Identifies a binary data header or a binary data trailer. Set to X'40'.
C02	X[1]	Record identifier	Identifies a binary data trailer when the dividing identifier value is set to X'40'. Set to X'54'.
D03	9[5]	Sequence number	Set to the same value as the value of the sequence number of the corresponding binary data header. Described in five standard numeric characters.
H04	9[4]	Relating number	Set to the same value as the value of the relating number of the corresponding binary data header.
T05	Bin32	Effective length of the last bit string storage area	Indicates the effective length of the large bit strings used to hold EDI user data and stored left-justified in the last binary unit.
T06	Bin32	Total number of logical records	Indicates the total number of logical records forming binary data. The binary data header, all binary units and the binary data trailer are counted in.
F41	X[232]	Reserved area 1	Filled with X'20'.

5. Relating number functions



Message group
0001
0001
Message
Binary data
Relate message to binary data by assigning the same relating number to them.
(Message can be related to binary data in a different message group in the same way.)

Fig. 6-5 Relating number functions

Annex 7 (Normative):
Data elements and format of special messages

1. Data elements and format of receive acknowledge message

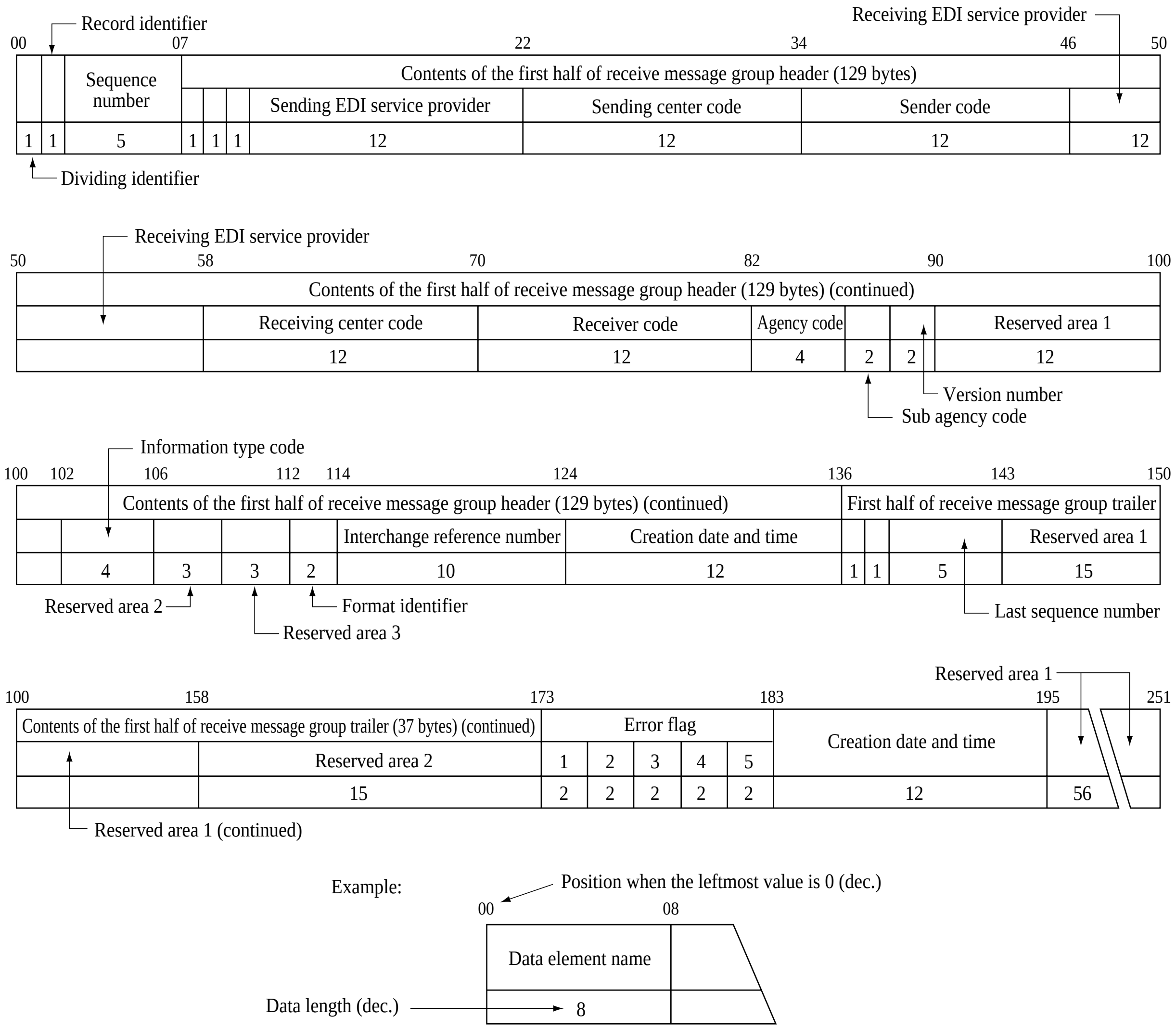

Record identifier
Receiving EDI service provider
00
07
22
34
46
50
Sequence number
Contents of the first half of receive message group header (129 bytes)
Sending EDI service provider
Sending center code
Sender code
1
1
5
1
1
1
12
12
12
12
Dividing identifier
Receiving EDI service provider
50
58
70
82
90
100
Contents of the first half of receive message group header (129 bytes) (continued)
Receiving center code
Receiver code
Agency code
Reserved area 1
12
12
4
2
2
12
Version number
Sub agency code
Information type code
100
102
106
112
114
124
136
143
150
Contents of the first half of receive message group header (129 bytes) (continued)
First half of receive message group trailer
Interchange reference number
Creation date and time
Reserved area 1
4
3
3
2
10
12
1
1
5
15
Reserved area 2
Format identifier
Reserved area 3
Last sequence number
Reserved area 1
100
158
173
183
195
251
Contents of the first half of receive message group trailer (37 bytes) (continued)
Error flag
Creation date and time
Reserved area 2
1
2
3
4
5
15
2
2
2
2
2
12
56
Reserved area 1 (continued)
Example:
Position when the leftmost value is 0 (dec.)
00
08
Data element name
Data length (dec.)
8

Fig. 7-1 Data element format of receive acknowledge message

Table 7-1 Data elements of receive acknowledge message

Symbol	Data type (length)	Data element	Notes (value to set)		
C01	X[1]	Dividing identifier	Set to X'39'.		
C02	X[1]	Record identifier	Set to X'44'.		
D03	9[5]	Sequence number	Indicates the sequence in which messages are arranged in one message group. The first sequence number 1 is assigned to the message immediately after the message group header and the following messages are numbered in ascending order. Described in five standard numeric characters.		
E51	129 bytes	Contents of the first half of receive message group header (129 bytes)	C01	X[1]	Dividing identifier
			C02	X[1]	Record identifier
			C03	X[1]	Operation mode
			C04	X[12]	Sending EDI service provider
			C05	X[12]	Sending center code
			C06	X[12]	Sender code
			C07	X[12]	Receiving EDI service provider
			C08	X[12]	Receiving center code
			C09	X[12]	Receiver code
			C10	X[4]	BPID agency code
			C11	X[2]	BPID sub agency code
			C12	X[2]	BPID version number
			F11	X[12]	Reserved area 1
			C14	X[4]	Information type code
			C15	9[3]	Reserved area 2
			C16	9[3]	Reserved area 3
			C17	X[2]	Format identifier
			C18	X[10]	Interchange informative number
			C19	X[12]	Creation date and time
E52	37 bytes	Contents of the first half of receive message group trailer (37 bytes)	C01	X[1]	Dividing identifier
			C02	X[1]	Record identifier
			E03	9[5]	Last sequence number
			E04	9[15]	Reserved area 1
			E05	9[15]	Reserved area 2
E55	9[2]	Error flag 1	An error detected by the translator on the receiving side. Described in two standard numeric characters. (Note)		
E56	9[2]	Error flag 2	Same as above		
E57	9[2]	Error flag 3	Same as above		
E58	9[2]	Error flag 4	Same as above		
E59	9[2]	Error flag 5	Same as above		
E60	9[12]	Creation date and time	The message creation date and time are described in the YYMMDDHHMMSS format using standard numeric characters.		
F61	X[56]	Reserved area 1	Filled with X'20'.		

Note) Two space characters can be used instead of "00."

2. Data elements and format of error message

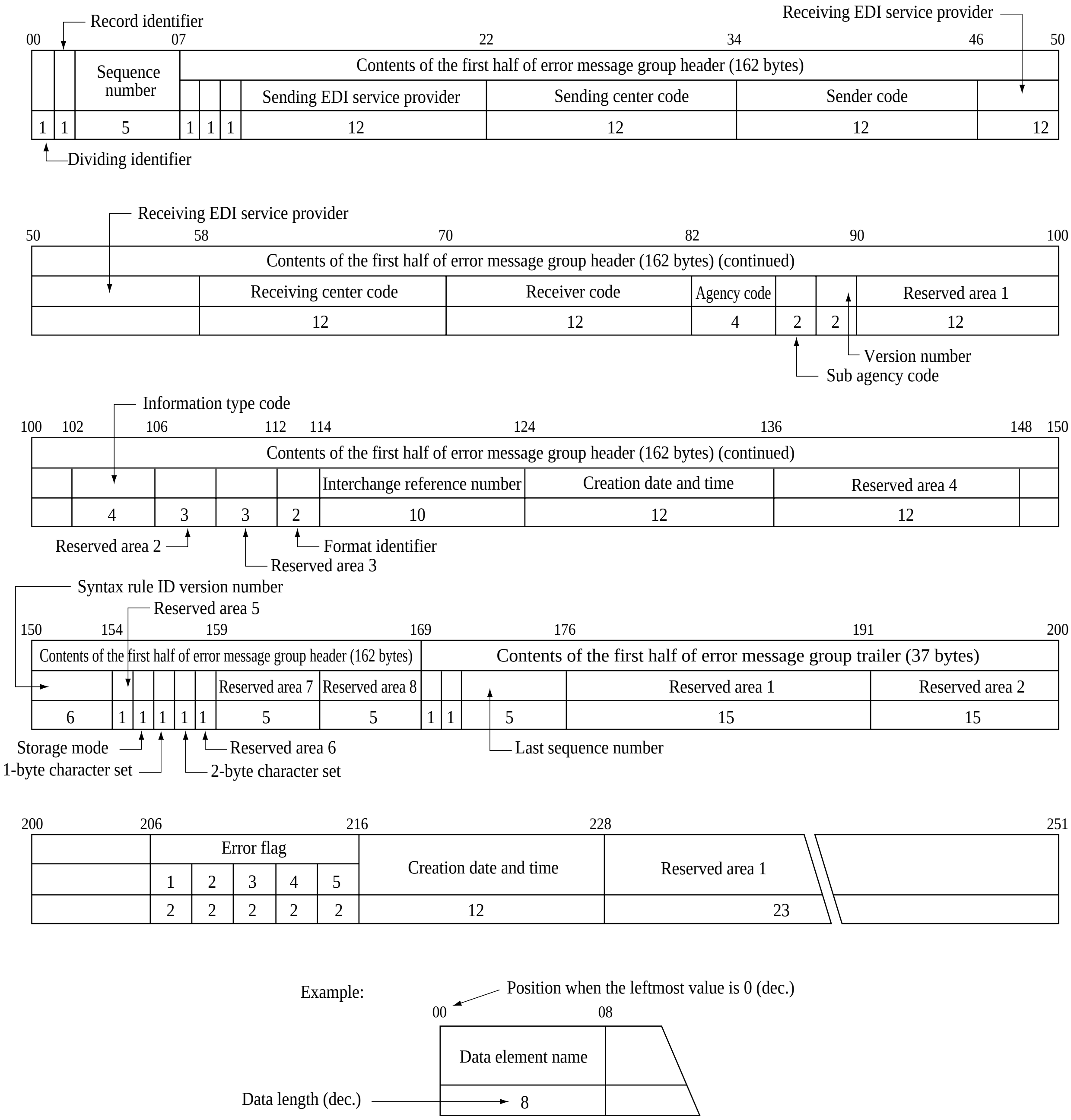


Fig. 7-2 Data element format of error message

Table 7-2 Data elements of error message

Symbol	Data type (length)	Data element	Notes (value to set)		
C01	X[1]	Dividing identifier	Set to X'39'.		
C02	X[1]	Record identifier	Set to X'44'.		
D03	9[5]	Sequence number	Indicates the sequence in which messages are arranged in one message group. The first sequence number 1 is assigned to the message immediately after the message group header and the following messages are numbered in ascending order. Described in five standard numeric characters.		
E71	162 bytes	Contents of the first half of error message group header (162 bytes)	C01	X[1]	Dividing identifier
			C02	X[1]	Record identifier
			C03	X[1]	Operation mode
			C04	X[12]	Sending EDI service provider
			C05	X[12]	Sending center code
			C06	X[12]	Sender code
			C07	X[12]	Receiving EDI service provider
			C08	X[12]	Receiving center code
			C09	X[12]	Receiver code
			C10	X[4]	BPID agency code
			C11	X[2]	BPID sub agency code
			C12	X[2]	BPID version number
			F11	X[12]	Reserved area 1
			C14	X[4]	Information type code
			C15	9[3]	Reserved area 2
			C16	9[3]	Reserved area 3
			C17	X[2]	Format identifier
			C18	X[10]	Interchange informative number
			C19	X[12]	Creation date and time
			F12	X[1]	Reserved area 4
			C21	X[6]	Syntax normative ID version number
			C22	X[1]	Reserved area 5
			C23	X[1]	Storage mode
			C24	X[1]	1-byte character set
			C25	X[1]	2-byte character set
			C26	X[1]	Reserved area 6
			C27	9[5]	Reserved area 7
			C28	9[5]	Reserved area 8
E72	37 bytes	Contents of the first half of error message group trailer (37 bytes)	C01	X[1]	Dividing identifier
			C02	X[1]	Record identifier
			E03	9[5]	Last sequence number
			E04	9[15]	Reserved area 1
			E05	9[15]	Reserved area 2
E75	9[2]	Error flag 1	Detected error. Described in two standard numeric characters (Note).		
E76	9[2]	Error flag 2	Same as above		
E77	9[2]	Error flag 3	Same as above		
E78	9[2]	Error flag 4	Same as above		
E79	9[2]	Error flag 5	Same as above		
E80	9[12]	Creation date and time	The message creation date and time are described in the YYMMDDHHMMSS format using standard numeric characters.		
F81	X[23]	Reserved area 1	Filled with X'20'.		

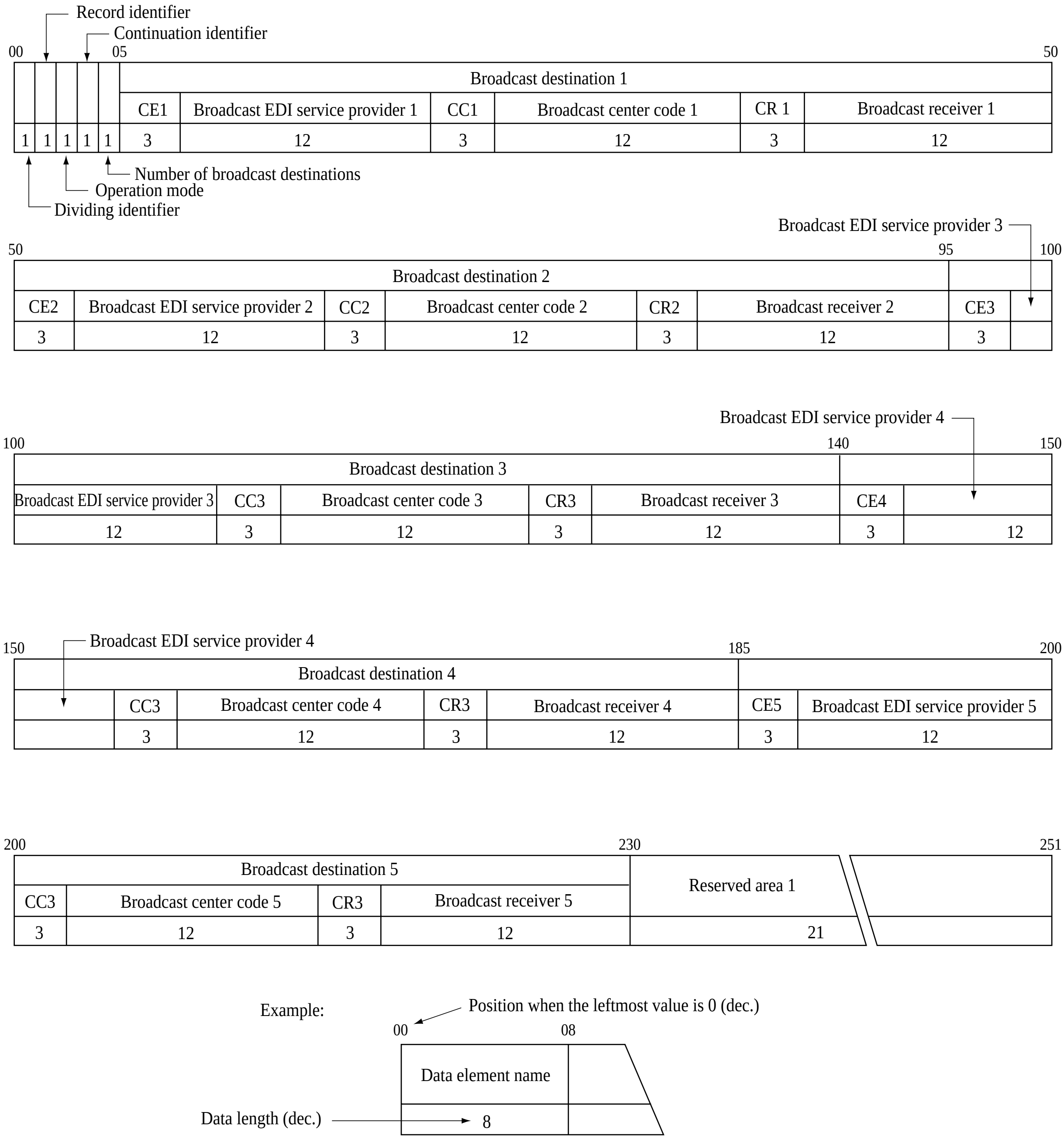
Note) Two space characters can be used instead of "00".

3. Error flag values (informative)

Table 7-3 Error codes for receive acknowledge/error messages

Error code	Error content
(Blank)	No error
00	No error
01	Not designated information type code (Whether or not code is output depends on the translator structure.)
02	Message group header not found.
03	Message group trailer not found.
04	Illegal syntax ID
05	Dividing identifier sequence error
10	Detection of undefined control tag
11	Detection of illegal data tag
12	Multi detail header not found in execution format conversion table.
13	Multi detail trailer not found in execution format conversion table.
14	Number of repeatings on local side (standard side) exceeds that on standard side (local side).
15	Data length exceeds maximum value.
16	Value of data for error detection not numeric.
17	Value for numerical conversion not numeric.
18	Data on standard side is longer than data on local side.
19	Record identifier not X'44' (no message found).
20	Message too long
21	No end of TFD area (X'FE').
22	Negative data exists (with 9 attribute, etc.).
30	Sequence numbers not in ascending order.
31	Error detected by error detection data
32	Execution format conversion table cannot be searched.
33	Illegal character code detected.
36	Illegal Y attribute (date) data
81	Interchange error
82	Broadcast error
99	Other errors

Annex 8 (Normative):
Data elements and format of broadcast header


Record identifier
Continuation identifier
00
05
50
Number of broadcast destinations
Operation mode
Dividing identifier
Broadcast EDI service provider 3
50
95
100
Broadcast EDI service provider 4
100
140
150
Broadcast EDI service provider 4
150
185
200
200
230
251
Example:
Position when the leftmost value is 0 (dec.)
00
08
Data length (dec.)

					Broadcast destination 1					
					CE1	Broadcast EDI service provider 1	CC1	Broadcast center code 1	CR 1	Broadcast receiver 1
1	1	1	1	1	3	12	3	12	3	12

Broadcast destination 2							
CE2	Broadcast EDI service provider 2	CC2	Broadcast center code 2	CR2	Broadcast receiver 2	CE3	
3	12	3	12	3	12	3	

Broadcast destination 3						
Broadcast EDI service provider 3	CC3	Broadcast center code 3	CR3	Broadcast receiver 3	CE4	
12	3	12	3	12	3	12

	Broadcast destination 4					
	CC3	Broadcast center code 4	CR3	Broadcast receiver 4	CE5	Broadcast EDI service provider 5
	3	12	3	12	3	12

Broadcast destination 5				Reserved area 1	
CC3	Broadcast center code 5	CR3	Broadcast receiver 5		
3	12	3	12	21	

Data element name	
8	

Note) CEn: Controlling agency code for the receiving EDI service provider n
CCn: Controlling agency code for the receiving center code n
CRn: Controlling agency code for the receiver code n
n: 1 to 5

Fig. 8-1 Data element format of broadcast header

Table 8-1 Data elements of broadcast header

Symbol	Data type (length)	Data element	Notes (value to set)
C01	X[1]	Dividing identifier	Set to X'30'.
C02	X[1]	Record identifier	Set to X'42'.
C03	X[1]	Operation mode	X'20' or X'30': normal message, X'31': test message
B03	X[1]	Continuation identifier	X'45': No continuation broadcast header follows., X'43': Another broadcast header follows.
B04	X[1]	Number of broadcast destinations	Indicates the number of broadcast destinations in the header. Described in standard numeric characters.
B11	X[3]	CE1	A code described in three limited standard characters
B12	X[12]	Receiving EDI service 1	A code described in twelve limited standard characters
B13	X[3]	CC1	A code described in three limited standard characters
B14	X[12]	Broadcast center code 1	A code described in twelve limited standard characters
B15	X[3]	CR1	A code described in three limited standard characters
B16	X[12]	Receiver code 1	A code described in twelve limited standard characters
B21	X[3]	CE2	A code described in three limited standard characters
B22	X[12]	Receiving EDI service 2	A code described in twelve limited standard characters
B23	X[3]	CC2	A code described in three limited standard characters
B24	X[12]	Broadcast center code 2	A code described in twelve limited standard characters
B25	X[3]	CR2	A code described in three limited standard characters
B26	X[12]	Receiver code 2	A code described in twelve limited standard characters
B31	X[3]	CE3	A code described in three limited standard characters
B32	X[12]	Receiving EDI service 3	A code described in twelve limited standard characters
B33	X[3]	CC3	A code described in three limited standard characters
B34	X[12]	Broadcast center code 3	A code described in twelve limited standard characters
B35	X[3]	CR3	A code described in three limited standard characters
B36	X[12]	Receiver code 3	A code described in twelve limited standard characters
B41	X[3]	CE4	A code described in three limited standard characters
B42	X[12]	Receiving EDI service 4	A code described in twelve limited standard characters
B43	X[3]	CC4	A code described in three limited standard characters
B44	X[12]	Broadcast center code 4	A code described in twelve limited standard characters
B45	X[3]	CR4	A code described in three limited standard characters
B46	X[12]	Receiver code 4	A code described in twelve limited standard characters
B51	X[3]	CE5	A code described in three limited standard characters
B52	X[12]	Receiving EDI service 5	A code described in twelve limited standard characters
B53	X[3]	CC5	A code described in three limited standard characters
B54	X[12]	Broadcast center code 5	A code described in twelve limited standard characters
B55	X[3]	CR5	A code described in three limited standard characters
B56	X[12]	Receiver code 5	A code described in twelve limited standard characters
F23	X[21]	Reserved area 1	Filled with X'20'.

Note)		
	CEn:	Controlling agency code for the receiving EDI service provider n
	Receiving EDI service n:	Receiving EDI service provider n
	CCn:	Controlling agency code for the receiving center code n
	CRn:	Controlling agency code for the receiver code n
	n:	1 to 5

Annex 9 (Normative):
Implementation standards

In developing a translator based on this standard, implementation of the following specifications is optional.

- Generation and analysis of implicit repeating of single user TFDs
- Value setting and analysis of data elements of message group header, C30, C31, C32, C33, C34 and C35
- Generation of error messages
- Generation and analysis of broadcast header

Comments of Components for Interchange Structure

Foreword

This section explains the contents of this document and is annexs and related matters, and does not form part of the specifications.

1. Background to the formulation

The Electronic Industries Association of Japan, which started standardizing EDI in 1987, developed a new EDI standard in cooperation with the Center for the Informatization of Industry (CII) in 1988. This EDI standard adopted a variable length format already employed in the USA, taking into account the characteristic message structure used in the electronics industry that uses more data elements than the distribution industry. As a result of various studies, the conclusion was that a rational format should consist of syntax rules, a standard message and standard data items. This proposed format was the same as the ANSI X.12 or EDIFACT (ISO9735 ver.1) standard under discussion at the time.

It was therefore decided to develop new syntax rules. According to EIAJ's proposed revisions, the data tag type syntax rules proposed by the CII evolved in the process of changing the data elements delimitation from a delimiter to a length tag. Although these syntax rules were expected to allow the use of kanji data and able to be applied to other industries, they were to be applied to a single industry (electronics equipment industry) for the moment. Therefore, the expanded functions were not implemented and use of the rules was limited to within the electronics industry. The rules were therefore named the EIAJ syntax rules. EIAJ also developed standard messages and standard data elements based on the EIAJ syntax rules, which were known as EIAJ standard messages and EIAJ standard data elements.

Since trials in the fall of 1988 to confirm practicability give favorable results, the EIAJ syntax normative, EIAJ standard messages and EIAJ standard data elements were unified in April 1989 as the EDI standard for the electronics industry and known as the "EIAJ-EDI Standard 1A". This then became the EIAJ standard.

Subsequently, with the success of the EIAJ standard in the electronics industry, other industries began taking note, and those seeking to introduce the EIAJ standard grew in number. However, as already mentioned, the EIAJ syntax rules that form the basis of the EIAJ standard allowed for a part of the functions to be suspended on the assumption that for the time being the rules would be used only within the electronics equipment industry. This gave rise to the problem that the rules were not applicable to other industries. More specifically, the suspension of one particular function, extension of the number of available data elements to 240 or more (the EIAJ standard can accommodate up to 239 data elements) was what prevented the normative from being applied to other industries.

Use of the standard only within the electronics industry enables messages to be designed even for 239 data elements; however, 239 element types is insufficient to allow use of the standard in the manufacturing industries, for example. Therefore, based on the findings of the studies conducted between 1989 and 1990, adoption of the EIAJ syntax rules as a JIS standard was suspended.

For this reason, the CII decided to implement the extensions to functions that had been suspended. Taking into account the unexpected wider application of the EIAJ standard, the CII decided to improve the EIAJ syntax normative to rectify the failings that had already appeared.

Studies were started in 1990 by the CII, the original proposer of the EIAJ syntax rules. Adjustments to the EIAJ were made at the beginning of 1991. The new improved EIAJ syntax rules were to be called the "CII Syntax Rules" as distinct from the EIAJ syntax rules, and special care was given to assure compatibility between the new standard and the conventional EIAJ standard (upward compatible). Furthermore, serious consideration was given to compatibility with the international standard EDIFACT (ISO9735 version 3). However, another measure to ensure compatibility was to be considered, since complete compatibility including the internal message structure would be difficult to achieve. Thus the "CII Syntax Rules (tentative) Specification 1.00" were established on April 1, 1991. In addition to this, requests from the construction industry and other industries, who had resolved to adopt the CII Syntax Normative, were added to the specifications, and a translator was developed between 1991 and 1992. The "CII Syntax Rules 1.10" were published in August 1992. At this point, the petrochemical industry started to employ the CII Syntax Rules.

As of the end of 1997, the CII Syntax Rules were widely adopted as the industry standard by twenty industries including the manufacturing industry. Meanwhile, new versions were repeatedly introduced to satisfy new needs and version 3.00 is to be released by March 1998. Furthermore, in order to meet the demand for stabilization and wider use of the rules, it was decided to adopt the rules as a JIS standard. Thus the JIS standard is equal to the CII Syntax Rules version 3.00. As part of the standardization, the name of the rules was changed to the "Syntax Rules for Cross-Industry Information Interchange" but abbreviated to "CII Syntax Rules" as before.

2. Planning for Part 4

The draft of Part 4 (Security Functions) is now being prepared so that it may be incorporated in the JIS standard by the year 2000.

3. Compatibility with EDIFACT (ISO 9735)

Compatibility of these rules with EDIFACT is achieved by using a converter. Trials have been conducted with the converter and no problems have been found.

Using the file interchange system based on a combination of Part 1 and Part 2 of these rules allows cross conversion with a combination of Part 1, Part 2 and Part 8 of ISO 9735. Using an interactive transfer system based on a combination of Part 1 and Part 2 or Part 1 and Part 3 allows cross conversion with a combination of Part 1, Part 3 and Part 8 of ISO 9735.

Part 2: The Structure of Message Groups

1. Scope

This standard apply to the structure of message groups of electronic data interchanged between government agencies, between industries, and between government agencies and industries.

2. Normative references

The following standards are referred to in these rules. The latest version of each standard is used.

JIS X 0201: Code for information interchange

JIS X 0208: Code of the Japanese graphic character set for information interchange

JIS X 0221: Universal multiple-octet coded character set (UCS)

3. Notation

3.1 Description of data

a) Data lengths are given in bits or bytes (1 byte = 8 bits).

b) Data values are described in 1-byte character strings or 2-byte kanji strings or in numeric characters.

c) 1-byte character strings are graphic characters described in (JIS-X0201) or hexadecimal.

d) 2-byte character strings are kanji characters described in (JIS-X0208) or hexadecimal.

e) Numeric values are described in decimal/hexadecimal.

f) Decimal values are strings consisting of the numeric characters 0 to 9, and are described as character strings.

g) Hexadecimal values are consist of numeric characters from 0 to 9 and an alphabetic character (A, B, C, D, E or F). One byte is written, for example, as X'43'.

h) Binary data are strings of bits, unless otherwise indicated.

i) Unsigned binary values are given in hexadecimal. If the length (number of bits) of an unsigned binary is not a multiple of 8, zeroes should be added to the left of the value.

3.2 Data element symbols

In order to identify each data element, a data element name and a symbol name composed of numeric characters or a three-character string are assigned to it.

4. Structure of message group

This standard specifies to the structure of normal message groups. For the structure of short form message groups, see Part 3.

A message group contains message group components arranged according to the following rules.

a) Start of message group (leftmost position)

A message group header must be placed at the start of the message group.

b) End of message group (rightmost position)

A message group trailer must be placed at the end of the message group.

c) Internal message group structure

The following components can be placed between a message group header and a message group trailer.

1) Security header message

This message must be placed immediately after the message group header. Only one security header message can be included in one message group. A security trailer message must be placed in the message group that contains a security header message.

2) Transaction message

Multiple transaction messages can be placed in one message group.

3) Receive acknowledge message

Multiple receive acknowledge messages can be placed in one message group.

4) Error message

Multiple error messages can be placed in one message group.

5) Sub security header message

Multiple sub security header messages can be placed in one message group. Transaction messages, receive acknowledge messages, error messages or binary data must be placed immediately after this message.

6) Security trailer message

This message must be placed immediately before a message group trailer. Only one security trailer message can be included in one message group. A security header must be placed in the message group containing a security trailer message.

7) Binary data

Multiple binary data can be placed in one message group.

d) Prohibited structures

The following structures are prohibited.

1) Both a receive acknowledge message and a transaction message are contained in the same message group.
2) Both a receive acknowledge message and an error message are contained in the same message group.
3) Both a receive acknowledge message and binary data are contained in the same message group.
4) Both an error message and a transaction message are contained in the same message group.
5) Both an error message and binary data are contained in the same message group.

e) Zero message

This message is composed of a message group header and a message group trailer.

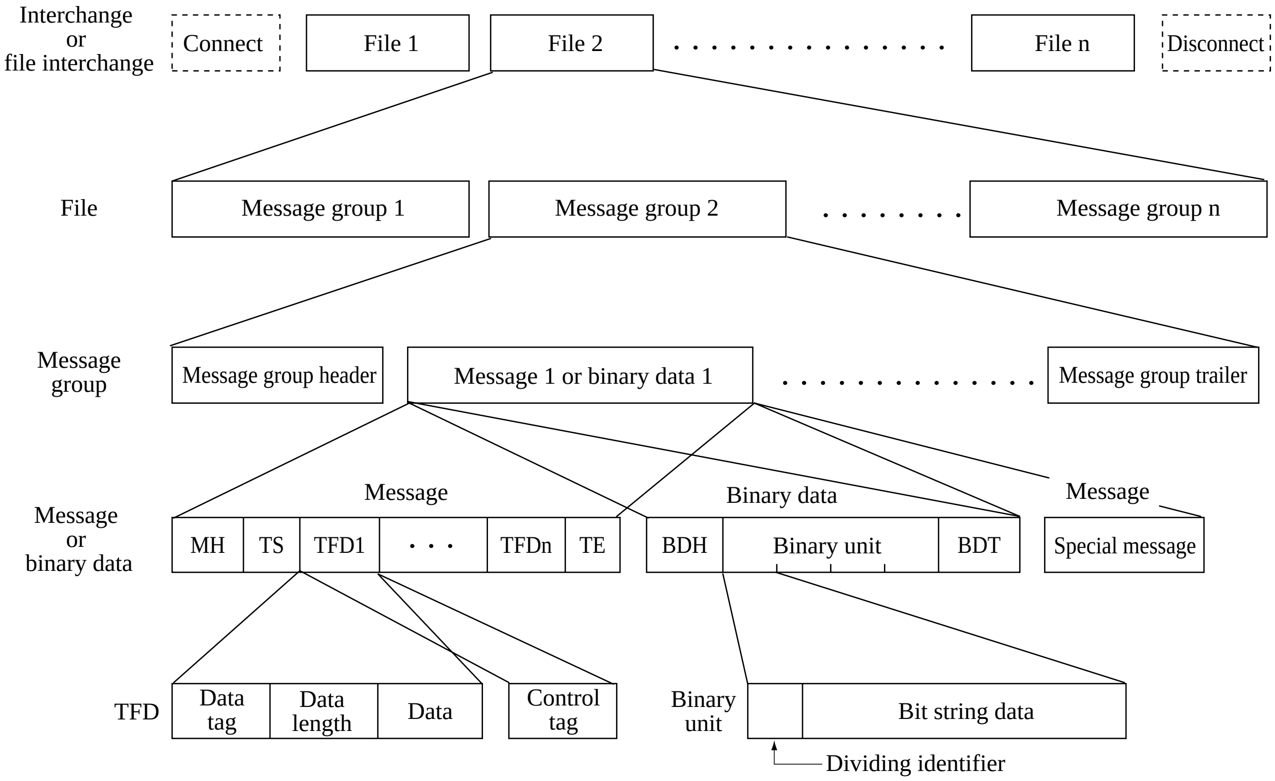
f) Broadcast header

A broadcast header must be placed immediately before a message group header.

5. Structure of application message

An application message contains seven message group components: a message header, a security header message, a sub security header message, a transaction message, binary data, a security trailer message and a message group trailer.

a) A security header message is paired with a security trailer message to secure the whole message group.
b) A sub security header message is used to secure one transaction message or one string binary data placed immediately after it. When no transaction message or binary data follows the sub security header message, the message is invalid.
c) A message group not containing a transaction message or binary data is a zero operation message (see Section 6).



Note) MH: Message header TS: Start of TFD area
TE: End of TFD area
BDH: Binary data header BDT: Binary data trailer

Fig. 1 Basic structure of message group (Using file transfer system)

Table 1 Message processing components

Component	Broadcast header	Message group header	Security header	Sub security header	Transaction message	Binary data	Security trailer	Message group trailer
Required/optional	○	●	△	○*1	○*2	○*2	△	●

Note) ● : Required ○ : Optional (can be omitted)
△ : Optional. Security header and security trailer must be paired.
○*1 : Transaction message or binary data must follow directly after the header.
○*2 : A message with both transaction message and binary data omitted from a processing message is an operation message.

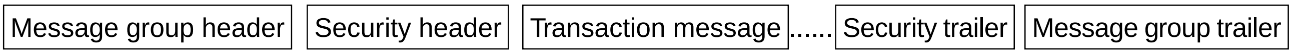


Fig. 2 Sample message processing structure

6. Structure of operation message

An operation message contains seven message group components: a message group header, a security header message, a sub security header message, a receive acknowledge message, an error message, a security trailer message and a message group trailer.

a) A security header message is paired with a security trailer message to secure the whole message group.

b) A sub security header message is used to secure one receive acknowledge message or error message placed immediately after it. This message is invalid if no receive acknowledge message or error message follows the sub security header message.

c) A receive acknowledge message and an error message cannot both be included in the same message group.

d) A message group that does not contain a receive acknowledge message or an error message is a zero operation message.

e) An error message is used by the EDI service provider to inform the customer (user) of an error that occurs during EDI service processing.

Table 2 Components of operation message

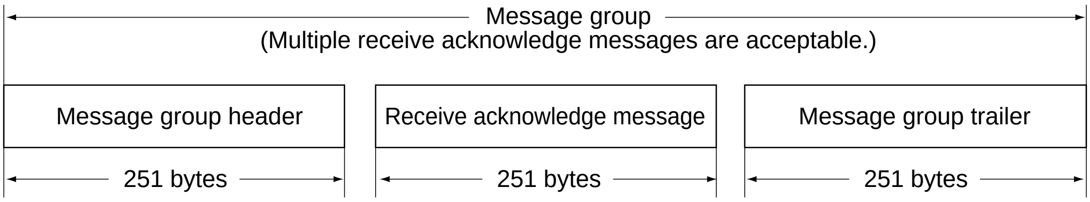
Component	Broadcast header	Message group header	Security header	Sub security header	Transaction message	Binary data	Security trailer	Message group trailer
Required/optional	○	●	△	○*1	◎*2	◎*2	△	●

Note) ● : Required ○ : Optional (can be omitted)

△ : Optional. Security header and security trailer must be paired.

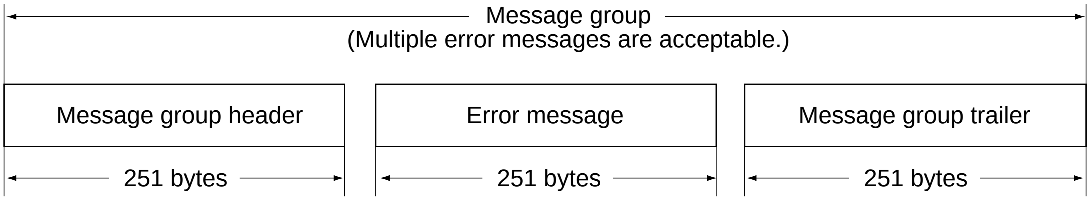
○*1 : Receive acknowledge message or error message must follow directly after the header.

◎*2 : A receive acknowledge message and an error message cannot both be included in the same message group. An operation message that does not contain a receive acknowledge message or an error message is an zero message.


Message group
(Multiple receive acknowledge messages are acceptable.)
Message group header
Receive acknowledge message
Message group trailer
251 bytes
251 bytes
251 bytes

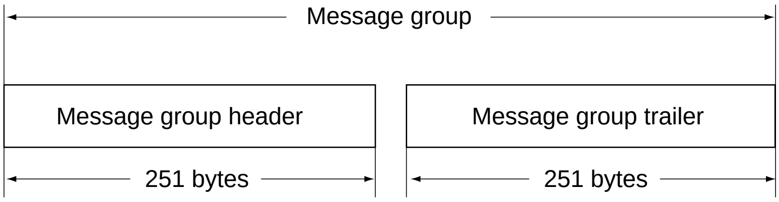
(1) Information type code of message group header .... 9001 (X'39303031')
(2) Format type of message group header....................................... 20 (X'3230')
(3) Sequence number of receive acknowledge message .... normally 00001 (X'3030303031')
(4) Last sequence number of message group trailer .... normally 00001 (X'3030303031')

Fig. 3 Sample receive acknowledge message structure



(1) Information type code of message group header .... 9201 (X'39323031')
(2) Format type of message group header .... 20 (X'3230')
(3) Sequence number of error message .... normally 00001 (X'3030303031')
(4) Last sequence number in message group trailer .... normally 00001 (X'3030303031')

Fig. 4 Sample error message structure



(1) Information type code of message group header .... 9101 (X'39313031')
(2) Format type of message group header .... 20 (X'3230')
(3) Last sequence number of message group trailer .... normally 00001 (X'3030303031')

Fig. 5 Sample zero operation message structure

7. Structure of broadcast message

A broadcast message contains a broadcast header and a message group.

a) A broadcast header indicates that the message group immediately after the header is a broadcast message. If a broadcast header is not followed by a message group, it is invalid.

b) One broadcast header designates five destinations maximum. Multiple broadcast headers can be used to designate six or more destinations. A continuation identifier (B03) in the broadcast header is used to indicate the existence of another broadcast header. When the value of the continuation identifier (B03) in a broadcast header is set to X'43'. the identifier indicates that another broadcast header exists, and when the value is set to X'45'. it indicates that no broadcast header follows.

   1) Continuation identifier (B03) value = X'43'............. header(s) continue

   2) Continuation identifier (B03) value = X'45'............. no header

c) A broadcast header is used by the original sender to inform the EDI service provider for broadcast processing that the message following the header is a broadcast message. The broadcast header is deleted during the broadcast processing by the EDI service provider. Then the broadcast message is converted into a normal message group and sent to the final sender.

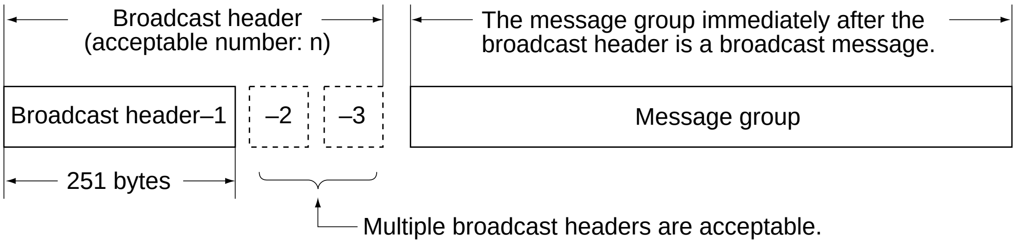

Broadcast header
(acceptable number: n)
The message group immediately after the broadcast header is a broadcast message.
Broadcast header–1
–2
–3
Message group
251 bytes
Multiple broadcast headers are acceptable.

Fig. 6 Sample broadcast message structure

8. Storage structure

Two modes are provided: dividing variable length mode and dividing fixed length mode.

8.1 Basic rules

a) The same storage structure must be used for one message group (including related broadcast headers).

b) Values for the format identifier (C17) and the storage mode (C23) in the message group header must be set as follows to identify the storage structure.

   1) Format identifier (C17)

      - X'3130' .... Processing message in dividing variable length mode
      - X'3131' .... Processing message in dividing fixed length mode
      - X'3230' .... Receive acknowledge message and error message (except for zero operation message) in the dividing variable length mode and the dividing fixed length mode.

   2) Storage mode (C23)

      - X'53' .... Dividing variable length mode
      - X'4D' or X'20' .... Dividing fixed length mode

c) The dividing variable length mode is used to store a message group (including related broadcast header) in a variable length record file. The dividing fixed length mode is used to store a message group (including related broadcast header) in a fixed length record file or a non-structured file (undefined length file).

d) In the dividing variable length mode, a variable length physical record with a maximum record length of 32001 bytes or below is used, and in the dividing fixed length mode, a 251-byte fixed length physical record is used.

8.2 Dividing variable length mode

In the dividing variable length mode, a message group (including related broadcast header) is stored into a physical record according to the following rules.

a) As for all message group components (including broadcast header) except for transaction messages, one component with a length of 251 bytes must be stored in one variable length physical record.

   For binary units of binary data, the dividing variable length mode must be selected. They must be stored in a variable length physical record whose maximum length is 32001 bytes (including the dividing identifier).

b) A transaction message must be divided into arbitrary-length segments and each segment (including the dividing identifier) must be stored in a variable length physical record whose maximum length is 32001 bytes. The following rules must be observed.

   1) Dividing must be performed from the beginning (left end) of the transaction message to the end.

   2) The segments into which a transaction message is divided must be stored in variable length physical records in the same sequence as when dividing.

c) The length of the first segment must be 32001 bytes or below.

   The value of the dividing identifier (C01) placed at the beginning (left end) of the first segment must be set to X'31'. When a transaction message is not divided, the value must be set to X'39'.

d) The length of each segment except for the first one must be 32000 bytes or below (32001 bytes including the length of the dividing identifier).

   Place a dividing identifier at the beginning of each segment except for the first one and the value of the added dividing identifier must be set as follows.

   1) The value of the dividing identifier for the second segment must be set to X'32'. and the value of the dividing identifier for the third segment must be set to X'33'. For the 4th to 8th segments, values must be set to X'34', X'35', X'36', X'37' and X'38' in ascending order.

   2) The value of the dividing identifier added to the ninth segment must be set to X'31'. The sequence X'31' to X'38' is repeated.

   3) The value of the dividing identifier added to the last (rightmost) segment must be set to X'39' to identify the last segment.

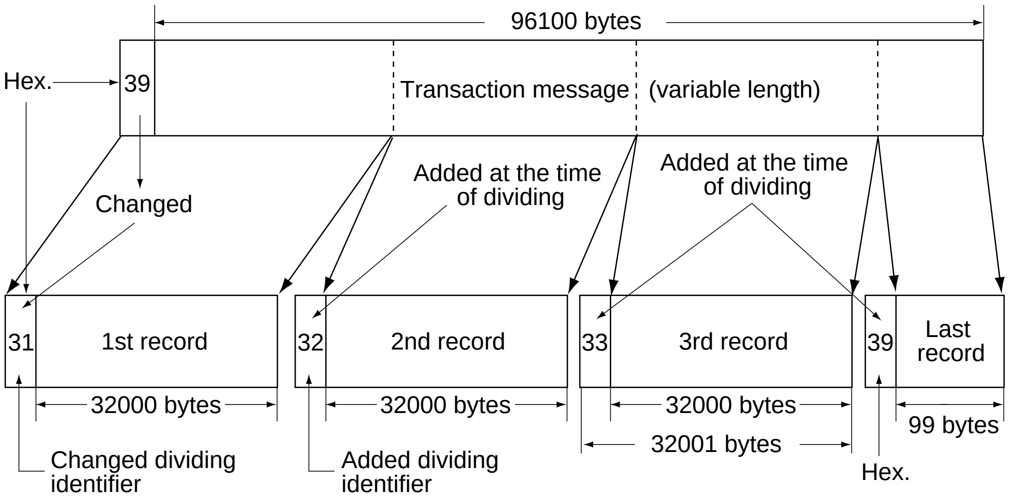

96100 bytes
Hex.
39
Transaction message (variable length)
Changed
Added at the time of dividing
Added at the time of dividing
31
1st record
32
2nd record
33
3rd record
39
Last record
32000 bytes
32000 bytes
32000 bytes
32001 bytes
99 bytes
Changed dividing identifier
Added dividing identifier
Hex.

Fig. 6 Dividing variable length mode (Example of dividing into four segments)

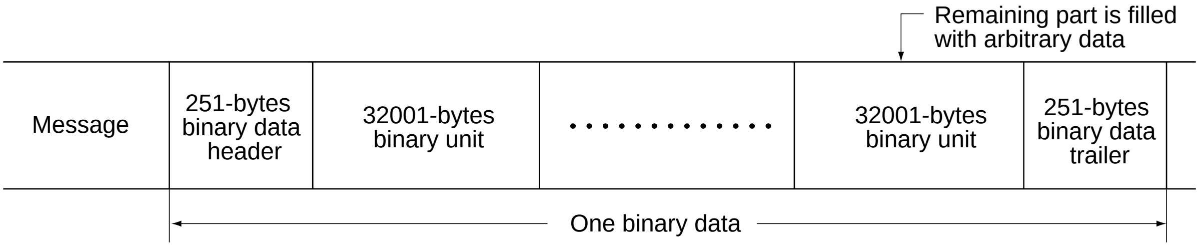

Remaining part is filled with arbitrary data
Message
251-bytes binary data header
32001-bytes binary unit
. . . .
32001-bytes binary unit
251-bytes binary data trailer
One binary data

Fig. 7 Storage structure for binary data in dividing variable length mode

8.3 Dividing fixed length mode

In the dividing fixed length mode, a message group (including related broadcast header) is stored in a physical record according to the following rules.

a) For all message group components (including broadcast header) except for a transaction message, one component with a length of 251 bytes is stored into one 251-byte fixed length physical record.

For binary units of binary data, the dividing fixed length mode must be selected and each binary unit is stored in a 251-byte fixed length physical record.

b) A transaction message is divided into fixed length segments, each having a length of 251 bytes (including the length of the dividing identifier). Each segment is stored in a fixed length logical record.

A transaction message must be divided and stored according to the following rules.

1) Dividing must be performed from the beginning (left end) of the transaction message to the end.

2) Each segment must be stored in a fixed length physical record in the same order as when dividing.

c) The length of the first segment must be 251 bytes. When the length of the first segment falls short of the fixed length (251 bytes), the right margin must be filled with X'20'.

The value of the dividing identifier (C01) placed at the beginning (left end) of the first segment must be set to X'31'. When the transaction message is not divided, the value must be set to X'39'.

d) The length of each segment except for the first one must be 250 bytes (251 bytes including the length of the dividing identifier). Because a variable length transaction message is divided into fixed length segments, the length of the last segment usually falls short of the fixed length (250 bytes). Therefore, the right margin must be filled with X'20'.

Place a dividing identifier at the beginning of each segment except for the first one and the value of the added dividing identifier must be set as follows.

1) The value of the dividing identifier added to the second segment must be set to X'32'. and that of the dividing identifier added to the third segment must be set to X'33'. For the 4th to 8th segments, values must be set to X'34', X'35', X'36', X'37' and X'38' in ascending order.

2) The value of the dividing identifier added to the ninth segment must be set to X'31'. In this way, the sequence from X'31' to X'38' is repeated.

3) The value of the dividing identifier added to the last (rightmost) segment is always set to X'39' to identify the last segment.

Table 3 Dividing a message in the dividing fixed length mode

Message length (D04) or extended message length (D06)	Number of records
1 to 250 251 to 500 501 to 750 ⋮	1 record 2 records 3 records ⋮
250*(n-1)+1 to 250*n	n records

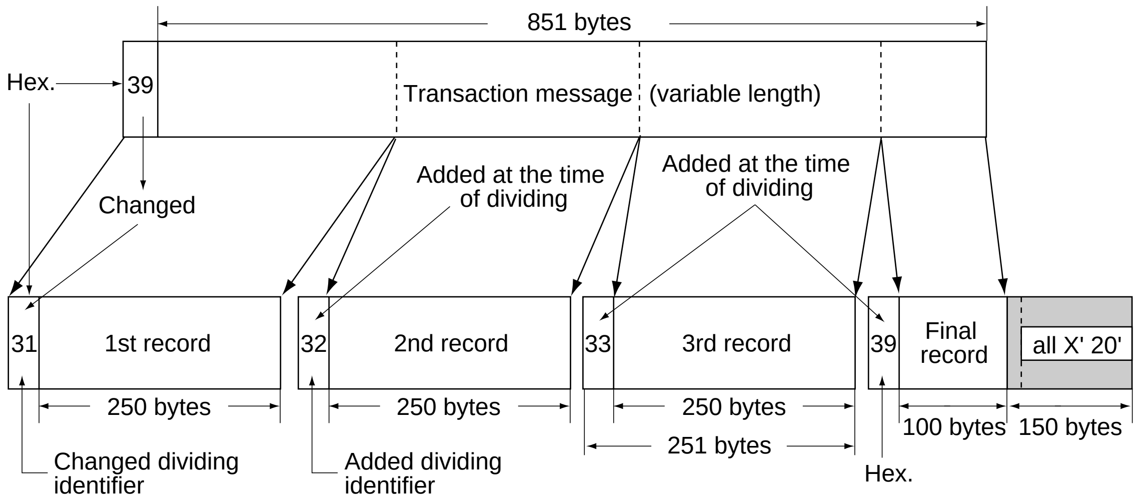


Fig. 8 Dividing fixed length mode (Example of dividing into four segments)

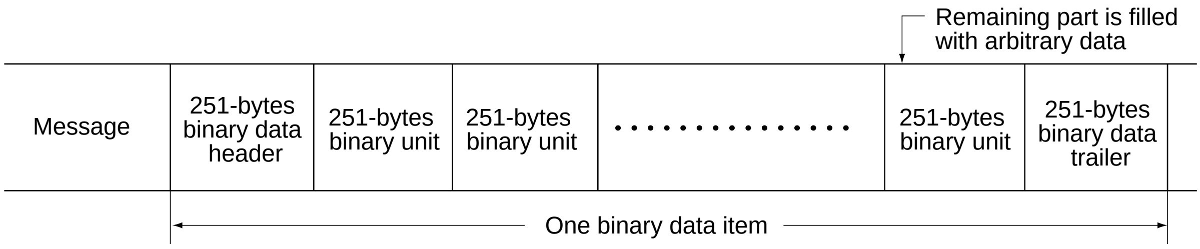


Fig. 9 Storage structure for binary data in dividing fixed length mode

Annex 1 (Normative): Information type code value of message group header

1. Application message

The value of the information type code (C14) in the message group header of a application message is fixed by the operation.

2. Operation message

The value of the information type code (C14) in the message group header of an operation message is set as follows.

- Receive acknowledge message .... 9001 (X'39303031')
- Error message .... 9201 (X'39323031')
- Zero message .... 9101 (X'39313031')

Annex 2 (Normative): Implementation standards

1. Packaging method

In order to develop a translator based on this standard, the standard must be combined with Part 1.

2. Optional functions

In developing a translator based on these rules, implementation of the following functions is optional.

a) Generation and interpretation of the security function and the related message group components
   - Generation and interpretation of security header
   - Generation and interpretation of sub security header
   - Generation and interpretation of security trailer

b) Generation and interpretation of the broadcast function and related message group components
   - Generation and interpretation of broadcast header

3. Recommended functions

In developing a translator based on this standard, it is recommended to implement the following functions related to operation messages.

- Generation and interpretation of receive acknowledge messages
- Generation and interpretation of error messages
- Generation and interpretation of zero messages

Annex 3 (Informative):
Relationship with telecommunications system

1. File transfer system

When this standard is applied to a file transfer system, interchange must be performed in units of a file where one or more message groups are stored.

2. Message transfer system

When this standard is applied to a message transfer system, interchange must be performed in units of one message group.

Part 3: The Structure for Short Form Message Groups

1. Scope

This standard applies to the structure of short form message groups of electronic data interchanged between government agencies, between industries, and between government agencies and industries.

2. Normative references

The following standards are referred to in these rules. The latest version of each standard is used.

JIS X 0201: Code for information interchange

JIS X 0208: Code of the Japanese graphic character set for information interchange

JIS X 0221: Universal multiple-octet coded character set (UCS)

3. Notation

3.1 Description of data

a) Data lengths are given in bits or bytes (1 byte = 8 bits).

b) Data values are described as a 1-byte character string or 2-byte kanji string or in numeric characters.

c) 1-byte character strings are graphic characters described in (JIS-X0201) or hexadecimal.

d) 2-byte character strings are kanji characters described in (JIS-X0208) or hexadecimal.

e) Numeric values are described in decimal/hexadecimal.

f) Decimal values are strings consisting of the numeric characters 0 to 9, and are described as character strings.

g) Hexadecimal values are consist of numeric characters from 0 to 9 and an alphabetic character (A, B, C, D, E or F). One byte is written, for example, as X'43'.

h) Binary data are strings of bits, unless otherwise indicated.

i) Unsigned binary values are given in hexadecimal. If the length (number of bits) of an unsigned binary is not a multiple of 8, zeroes should be added to the left of the value.

3.2 Data element symbols

In order to identify each data element, a data element name and a symbol name composed of numeric characters or a three-character string are assigned to it.

4. Structure of short form message groups

In case the value of short form message group identifier (C29) in a message group header is X'49', it means that the message group is short formed. The value of short form message group identifier (C29) is X'20' or X'53', the message group is normal. (See part 2)

A short form message group contains message group components arranged according to the following rules.

a) Start of message group (leftmost position)

A message group header must be placed at the start (leftmost position) of the message group.

b) Immediately after message group header

One of the following components must be placed immediately after the message group header.

- Sub security header message
- Transaction message
- Binary data
- Receive acknowledge message
- Error message

c) Immediately after sub security header message

One of the following components must be placed immediately after the sub security header message.

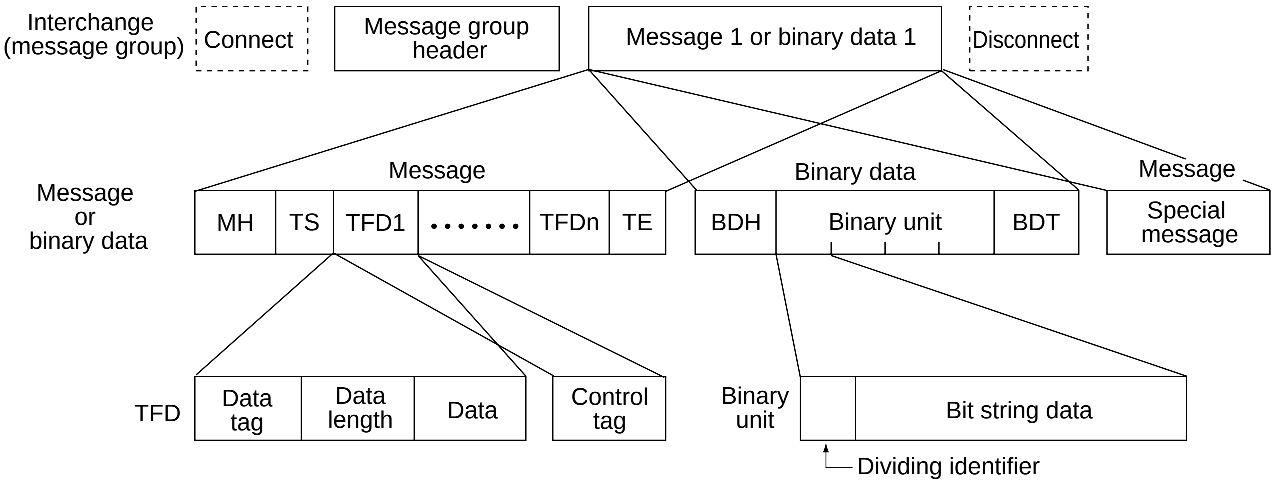
- Transaction message
- Binary data
- Receive acknowledge message
- Error message

5. Structure of short form application messages

A short form processing message contains five message group components: a message group header, a sub security header message, a transaction message and binary data.

a) When a short form processing message includes no sub security header message, it contains two message group components. When it includes a sub security header message, it contains three message group components.

b) A sub security header message is used to secure one transaction message or one string of binary data placed immediately after it.



Note) MH: Message header TS: Start of TFD area
TE: End of TFD area
BDH: Binary data header BDT: Binary data trailer

Fig. 1 Basic structure of short form message group (When a sub security message is not used.)

Table 1 Components of short form processing message

Components	Broadcast header	Message group header	Sub security header	Transaction message	Binary data
Required/optional	○	●	○^{*1}	○^{*2}	○^{*2}

Notes) ● : Required ○: Optional (can be omitted)
○^{*1}: Transaction message or binary data must follow directly after the header.
○^{*2}: Both transaction message and binary data cannot be included in the same short form processing message. The short form processing message which does not contain either a transaction message or a binary data is a short form zero operation message.

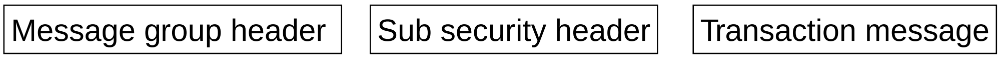


Fig. 2 Structure example of short form processing message

6. Structure of short form operation message

A short form operation message contains five message group components: a message group header, a sub security header message, a receive acknowledge message, and an error message.

a) When a short form operation message does not include a sub security header message, it contains two message group components. When it includes a sub security header message, it contains three message group components.

b) A sub security header message is used to secure one receive acknowledge message or one error message placed immediately after it.

c) A message group containing only one message group header is a zero operation message.

d) An error message is used by the EDI service provider to inform the customer (user) of an error that occurs during EDI service processing.

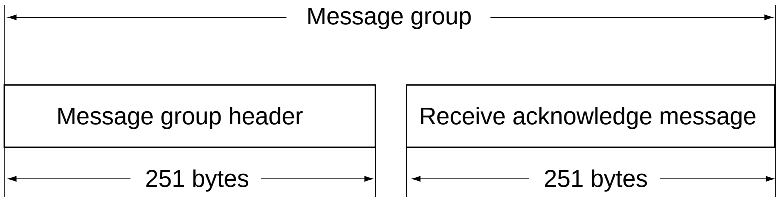
Table 2 Components of short form operation message

Components	Broadcast header	Message group header	Sub security header	Receive acknowledge message	Error message
Required/optional	○	●	○[*1]	◎[*2]	◎[*2]

Notes) ● : Required ○: Optional (can be omitted)

○[*1]: Receive acknowledge message or error message must follow directly after the header.

◎[*2]: A receive acknowledge message and an error message cannot both be included in the same short form operation message. A short form operation message that does not contain either a receive acknowledge message or an error message is a short form zero operation message.

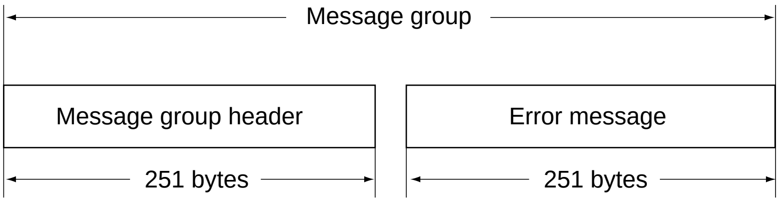


(1) Information type code of message group header .... 9001 (X'39303031')

(2) Format type of message group header....................................... 20 (X'3230')

(3) Sequence number of receive acknowledge message .... normally 00001 (X'3030303031')

Fig. 3 Sample short form receive acknowledge message structure

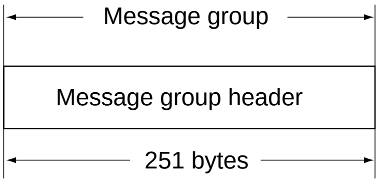


(1) Information type code of message group header .... 9201 (X'39323031')

(2) Format type of message group header.. 20 (X'3230')

(3) Sequence number of error message .... normally 00001 (X'3030303031')

Fig. 4 Sample short form error message structure



(1) Information type code of message group header .... 9101 (X'39313031')

(2) Format type of message group header.. 20 (X'3230')

Fig. 5 Sample short form zero message structure

7. Structure of broadcast header

A broadcast message contains a broadcast header and a short form message group.

a) A broadcast header indicates that the message group immediately after the header is a broadcast message. If a broadcast header is not followed by a message group, it is invalid.

b) One broadcast header designates five destinations maximum. Multiple broadcast headers can be used to designate six or more destinations. A continuation identifier (B03) in the broadcast header is used to indicate the existence of another broadcast header. When the value of the continuation identifier (B03) for a broadcast header is set to X'43'. the identifier indicates that another broadcast header exists, and when the value is set to X'45'. it indicates that no broadcast header follows.

   1) Continuation identifier (B03) value = X'43'.............header(s) continue

   2) Continuation identifier (B03) value = X'45'.............no header

c) A broadcast header is used by the original sender to inform the EDI service provider for broadcast processing that the message following the header is a broadcast message. The broadcast header is deleted during the broadcast processing by the EDI service provider. The broadcast message is then converted into a normal message group and sent to the final sender.

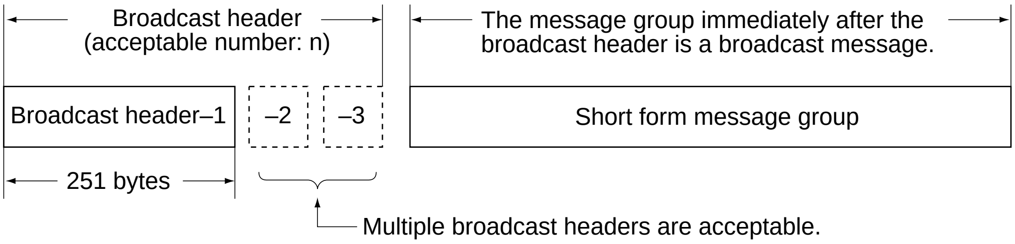

Broadcast header
(acceptable number: n)
The message group immediately after the broadcast header is a broadcast message.
Broadcast header–1
–2
–3
Short form message group
251 bytes
Multiple broadcast headers are acceptable.

Fig. 6 Sample broadcast message structure

8. Storage structure

The storage structure is the same as for a normal message group as specified in Section 8, Part 2.

Annex 1 (Normative): Information type code value of message group headers

Specified in Annex 1 of Part 2.

Annex 2 (Normative): Implementation standards

1. Packaging method

In order to develop a translator based on these rules, the rules must be applied in combination with the rules described in Part 1.

2. Optional functions

In developing a translator based on the rules described in Part 1 and Part 2, implementation of the functions relating to this normative (functions relating to short form message groups) is optional.

Annex 3 (Informative): Relationship with telecommunications system

1. File transfer system

When this standard is applied to a file transfer system, interchange must be performed in units of a file where one or more short form message groups are stored. Both a normal message group and a short form message can be stored in one file.

2. Message transfer system

When this standard is applied to a message transfer system, interchange must be performed in units of short form message groups.

Published: September 1998
Published by: Center for the Informatization of Industry
Kikai Shinko Kaikan Building 3-5-8, Shibakoen
Minato-ku, Tokyo, 105-0011 Japan
Printed by: Inter Group Corp.