e^2TP Message Specification

TEF950-S003-01.00.00/en

2005/03

T-Engine Forum

Contents

1. Introduction ---------- 1
1.1. Outline of this specification 1
1.1.1. TENeT architecture 1
1.2. Positioning of this specification 2
1.3. Scope of specification 2
1.4. Normative reference 2
1.5. Terminology 2
1.6. Endians 2

2. e2TP Messages ---------- 2
This section explains the format of e2TP messages and their contents ---------- 2
2.1. e2TP message format 2
2.1.1. eTRON ID 3
2.1.2. Message types 4
2.1.3. Thread ID 4

3. Mapping to ISO7816-4 APDU ---------- 5
3.1. APDU format 5
3.2. TENeT command format 5
3.3. TENeT response formats 6
3.3.1. On a normal processing 6
3.3.2. On an abnormal end 6
3.3.3. Response format containing multiple messages 7

1. Introduction

This specification prescribes the formats of the e^2TP messages and the method of mapping them to the ISO7816-4 APDU.

1.1 Outline of this specification

The TENeT is a framework designed to implement safe transactions of e-rights/e-money in a mobile environment. There are IC cards and mobile terminals on the TENeT. It is presupposed that an IC card inserted into a mobile terminal is used from an application programs on the mobile terminal.

The application issues operating instructions and responses by interchanging messages with the IC card. For this interchange of messages, the TENeT adopts the message passing format. Therefore, the application and the IC card can start transacting with another party after sending out a message, without having to wait for a response from the remote party.

Moreover, when the distributed transparency type message delivery mechanism of the TENeT is employed, the application does not have to apply relay processing to output messages from the card.

This specification establishes the e^2TP message formats that are common for the messages delivered over the TENeT.

1.1.1 TENeT architecture

Figure 1 represents the TENeT architecture that delivers messages in the form of distributed transparency between IC cards and between IC cards and applications. The messaging library transmits and receives the e^2TP messages synchronously (waits for a response message to a transmitted message) or asynchronously (continues processing after transmitting a message, without waiting for a response message). When the application has conducted an asynchronous message transmission, it acquires a receiving message by an event listener registered in advance, similarly to other event-driven systems (GUI framework, etc.)

According to this architecture, when an IC card or application outputs an e^2TP message, the messaging library first accepts the message and transmits it to the destination. Therefore, a message which is output by an IC card with destination to another IC card is transmitted by the messaging library directly to the destination IC card. This eliminates the need for relay processing by an application.

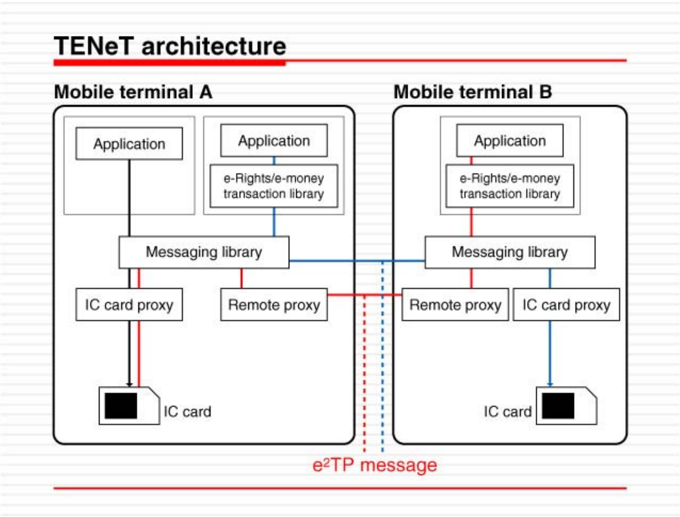

Figure 1. TENeT architecture

1.2 Positioning of this specification

To achieve the interoperability of IC cards and application programs that implement e-rights/e-money transactions, the TENeT has the following types of specifications prescribed:

- TENeT e-rights/e-money transaction API specification
  API designed to have the e-rights/e-money of an IC card managed and circulated from an application program.
- TENeT messaging API specification
  API designed to transmit/receive e^2TP messages to/from an application program.
- Extended eTP (e^2TP) message specification (This document)
  Formats of e^2TP messages and method of mapping them to the ISO7816-4 APDU.
- TENeT message specification
  e^2TP message sets with which TENeT-enabled IC cards should be provided.

1.3 Scope of specification

This document specifies the formats of e^2TP messages and the method of mapping of those messages to the ISO7816-4 APDU for their input/output to/from IC cards.

1.4 Normative reference

ISO/IEC Integrated circuit(s) cards with contacts

- Part 4: Interindustry commands for interchange, ISO/IEC 7816-4: 1995(E)

1.5 Terminology

- Table 1 defines the common terms employed in this document.

Term	Meaning
B	Abbreviation of Byte Indicates the byte.
MSB	Abbreviation of Most Significant Bit Indicates the highest-order bit of an array of bytes.

1.6 Endians

The network endians (big endians) are employed as endians for all of the values on the e^2TP message.

2. e^{2}TP Messages

This section explains the format of e^{2}TP messages and their contents.

2.1 e^{2}TP message format

To allow application programs to use the distributed processing between IC cards easily, the TENeT is provided with a mechanism designed to interchange messages in the form of distributed transparency between IC cards or between IC cards and application programs. In the TENeT, the messages delivered by means of this mechanism are called e^{2}TP (extended eTRON Transfer Protocol) messages. Table 2 gives the format of the e^{2}TP message.

The leading byte of the format value represents the version of the e^{2}TP message format. The version of this specification is 10h. The other values than that of the format are those specified when the specification version is 10h.

DestID and SrcID are the identifiers that indicate the destination and transmitting source of a message, respectively. The messaging library delivers the message to its destination according to this SrcID.

The TENeT messages specified in the "TENeT Message Specification" are input to the DATA section. The IC card discriminates a message according to it type, and applies processing to the DATA, which is the content of the message.

The thread ID, message types and eTRON ID will be explained in detail in the following subsections.

Table 2. e^{2}TP message format

Description		Size (Bytes)	Meaning
Format	Format	4	Indicates the version of the e2TP message format. Leading 1B: Specification version (10h) Remaining 3B: Reserved area (000000h)
DestID	Destination eTRON ID	16	eTRON ID of the destination of the message.
SrcID	Transmitting eTRON ID	16	eTRON ID of the transmitting source of the message.
ThreadID	Thread ID	20	ID indicative of the relation of the message.
MessageType	Message type	2	Type of message defined in each specification
LEN	Data length	2	Data length of the message to which a routing header is assigned.
DATA	Data	LEN	TENeT message

2.1.1 eTRON ID

eTRON ID is a 16-byte identifier defined in eTRON/16. The TENeT divides this eTRON ID into the former 12 bytes of "domain" and the latter 4 bytes of "port."

Each IC card on the TENeT is possessed by a proper domain. The IC card has port=0 as its own value, and therefore, the eTRON ID of the IC card is "domain|0."

A port is dispensed by the IC card uniquely at the request of an application. The application acquires, as its own identifier, "domain|port," which is a value consisting of a domain and port coupled. Because it is inefficient, from the viewpoint of system operation, for each application added to or deleted from a mobile terminal to possess its proper eTRON ID, a unique identifier is assigned to the application when such an ID is required, by way of operational solution.

Since ports are "expendable," there is the concern that ports may not be able to be dispensed any longer some day. However, approximately 4 billion ports can be assigned per domain. That is, it will take some 136 years to exhaust the ports, even if one ID is dispensed every second. Ports are practically inexhaustible.

2.1.2 Message types

The message type consists of a 2-byte value that classifies the TENeT messages specified in the "TENeT Message Specification." As for the breakdown of the message type, the higher one byte is a major classification that indicates a basic/exchange message, and the lower one byte, a minor classification indicating the message of the basic/exchange message. Details on each classification are given in Table 3 and Table 4, respectively.

Especially, the minor classification of messages among the basic/exchange messages classifies them into normal messages and error messages by the MSB of the lower one byte, and into TENeT messages by the digits of b6 on.

Table 3. Classification by higher one byte of message type

Major classification	b7	b6	b5	b4	b3	b2	b1	b0	16 Hexadecimal notation
TENeT message specification	0	x	x	x	x	x	x	x	-
Basic	0	0	0	0	0	0	0	0	00h
Exchange	0	0	0	0	0	0	0	1	01h
(RFU)	0	(Other digits)							-
Arbitrary	1	x	x	x	x	x	x	x	-

* When the MSB is 1, the message type can be used arbitrarily by AP.

Table 4. Classification by lower one byte of message type (Normal/error)

Major classification	b7	b6	b5	b4	b3	b2	b1	b0
Normal message	0	x	x	x	x	x	x	x
Error message	1	x	x	x	x	x	x	x

* When the MSB is 0, the message type is used for a normal message.
* When the MSB is 1, the message type is used for an error message.

2.1.3 Thread ID

The applications and IC cards establish correspondence in a series of e^2TP messages they input and output by means of a thread ID.

The thread ID is a value created by an application. When an application requests an operation from an IC card, a thread ID is assigned as an ID to identify that request uniquely. Since this thread ID is required to be unique only on an individual application basis, the application takes, as a thread ID, the value resulting from coupling the identifier 'serial,' which is unique for each operation request, to its own ID, namely, "domain|port|serial."

Take a file-creating message sent from an application to an IC card, for example. By assigning the same thread ID to the file-creating message and the response message to be returned from the IC card, the application can secure correspondence between the sending of the message and the response.

Moreover, in an exchange control, if an application instructs an IC card to switch, messages are interchanged several times between the application and the IC card. If all the messages are given the same thread ID, as in the case described above, the application can identify them as a series of related messages.

3. Mapping to ISO7816-4 APDU

This section explains the method for mapping the e^2TP messages defined in Section 2 to the APDU format, which is an IC card input/output format.

3.1 APDU format

ISO7816-4 is established as the international standard specification for IC card hardware and message formats [ISO7816-4]. IC cards conforming to this ISO7816-4 standard accepts only the messages compliant with the APDU (Application Protocol Data Unit) format. Figure 1 shows the APDU format. The values of CLA and INS classify the common commands specified in ISO7816-4. P1 and P2 are the parameters used by the common commands.

Header				Body		
CLA (1B)	INS (1B)	P1 (1B)	P2 (1B)	Lc (1B)	Data (Variable length)	Le (1B)

Figure 1. APDU format

e^2TP messages are made in the format defined by an original specification that does not conform to the ISO7816-4 standard. For interchange with IC cards, therefore, the TENeT employs the ISO7816-4 Envelope, so that IC cards conforming to ISO7816-4 can accept e^2TP messages. The routing header values specified in ISO7816-4 are assigned to the Envelope command/response. Therefore, it results that the format contains the e^2TP message specified in Subsection 2.1 inside the DATA of the Body section. (See Figure 2.)

3.2 TENeT command format

Figure 2 shows the ISO7816-4 Envelope command format mentioned in Subsection 3.1. The e^2TP message is accommodated in the Body section of the APDU format. Table 5 gives the prescribed values of the Envelope command.

Header				Body		
CLA (1B)	INS (1B)	P1 (1B)	P2 (1B)	Lc (3B)	Data (Variable length)	Le (2B)

C-APDU shall always be the Case 4 extension.

Message						
RoutingHeader						Payload
FORMAT (4B)	DestID (16B)	SrcID (16B)	ThreadID (20B)	Message Type(2B)	LEN (2B)	DATA (Variable length)

Figure 2. Command format

Table 5. Prescribed values of command format

Parameter	Value
CLA	00h (Fixed value)
INS	C2h (Fixed value)
P1	00h (Fixed value)
P2	00h (Fixed value)
Lc	Length of "Data" section inside the Body
Data	e2TP message

Le	0000h (Fixed value)

3.3 TENeT response formats

The format of the response that is output by an IC card after processing an Envelope command is discussed here. The response varies depending on whether the message has been processed normally, or the value of the Header section of the APDU command was illegal. The respective response formats are shown below.

3.3.1 On a normal processing

Figure 4 illustrates the normal response format. Table 6 shows the prescribed values of SW1 and SW2, the parameters.

Body	Trailer	
Data (Variable length)	SW 1 (1B)	SW 2 (1B)

Message						
RoutingHeader						Payload
FORMAT (4B)	DestID (16B)	SrcID (16B)	ThreadID (20B)	Message Type(2B)	LEN (2B)	DATA (Variable length)

Figure 4. Response format (on a normal end)

Table 6. Prescribed values of response format

Parameter	Value
Data	TENeT message
SW1	90h (Fixed value)
SW2	00h (Fixed value)

3.3.2 On an abnormal end

Figure 5 illustrates the format of the response issued when the processing ends abnormally due to an illegal routing header, such as mismatch between the Envelope command header or Lc and Le, and so on. Table 7 shows the prescribed values of SW1 and SW2, the parameters.

Trailer	
SW 1 (1B)	SW 2 (1B)

Figure 5. Response format (on an abnormal end)

Table 7. Prescribed values at the time of abnormal end

Description	Value	Reference
LEN,Lc,Le errors	6700h	The value of LEN, Lc or Le is illegal.
Command execution condition is not met.	6985h	Personalize incomplete
CLA error	6E00h	CLA of C-APDU is illegal.
INS error	6D00h	INS of C-APDU is illegal.
P1-P2 error	6A86h	P1 or P2 of C-APDU is illegal.
Ver error	6AA0h	Version of routing header is illegal.
SrcID error	6AA1h	SrcID of routing header is illegal.
DestID error	6AA2h	DestID of routing header is illegal.
LEN error	6AA3h	LEN of routing header is illegal.

(TBD)	(TBD)	(TBD)

3.3.3 Response format containing multiple messages

The TENeT sometimes sends the result of a processing to multiple remote parties simultaneously. In this case, multiple messages are first coupled together, and then, output to the Body section of the response message, as shown in Figure 6. The multiple messages that are output in this format do not necessarily arrive at the remote parties in the same order as they are accommodated in it.

This format can only be used for output from IC cards, being inhibited for the inputs of messages from applications to IC cards.

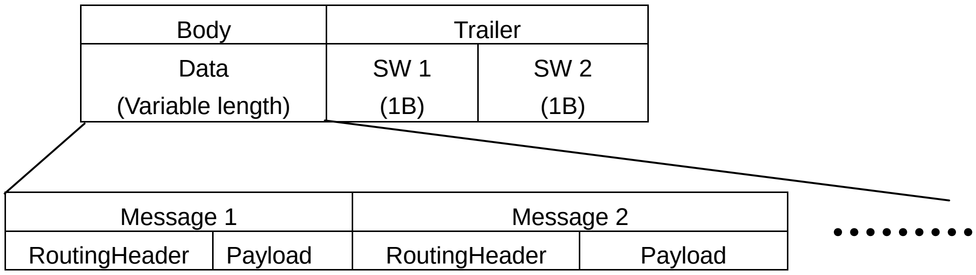

Figure 6. Response format containing multiple messages

[Bibliography]

- "TENeT Message Specifications, T-Engine Forum, 2005."