The drawaing contained in this Recommendation have been done in Autocad

10.2 *Operation of the TLMAU*

The section describes how the TLMAU will provide the TLM abstract service. The TLM abstract operations have been defined by abstract operations, sometimes with associated results or errors. These abstract operations, results and errors are realized via the exchange of TAPDUs between the TLM terminal and the TLMAU.

The realization of the abstract operations for the import and export ports linking the TLMAU and the MTS, is beyond the scope of this Recommendation. For the purpose of this section import and export operations will be considered to be similar to submission and delivery port operations.

10.2.1 *MessageSend*

The MessageSend operation will be provided by the TLMAU via the Send—, SendAck— and Exception— TAPDUs:

Upon receipt of Send—TAPDU by the TLMAU, the TLMAU will take the following actions:

1) The TLMAU will invoke the MTAS import abstract operation MessageSubmission with the following argument values:

Source of MTAS MessageSubmission arguments

MessageSubmission argument	CorrespondingSend—TAPDU component		Operation
	Element name	Value name	
originator—name	—	—	Authenticate User (remarque 1)
original—encoded—information—types	—	—	set by TLMAU to EITs of submitted IPM's body
content—type	—	—	set by TLMAU to “IPMS”
content—identifier	—	—	Local matter
content—correlator	—	—	Local matter
recipient—name	to, cc, bcc	primary—, copy—, blind—copy—recipient	constructed by TLMAU

Note 1 — Authenticate User is constructed from TID obtained from CSS terminal ID.

Note 2 — The IPM submitted as the content is constructed by the TLMAU. Send—TAPDU components representing IPMS elements are mapped onto the corresponding IPMS application protocol data unit (APDU) elements.

Note 3 — A multi—document messages will be submitted as an IP message with a multi—part body, each body part corresponding to a submitted document.

Note 4 — When this—IPM of IPMS element is omitted, the TLMAU shall construct this component which consists of the following components: originator name, Date and Time and, if necessary, a sequence number.

Other message submission arguments have a corresponding Send—TAPDU component. If this component is omitted, the default value applies.

2) If the MessageSubmission operation results in an error or if an error is detected in the Send—TAPDU, the TLMAU will return an Exception—TAPDU to the originating TLM terminal.

3) The TLMAU will, when required, return a SendAck—TAPDU to the originating TLM terminal following the successful completion of the MessageSubmission operation. The values of the SendAck—TAPDU will be set as follows:

Source of SendAck—TAPDU components

SendAck—TAPDU component		
Element name	Value name	Source
correlationInfo	call—id	CallIdentification that identifies previ Send—TAPDU being reported on
submissionId	submission—msg—id	MTS message—submission—identifie
submissionTime	submission—time	MTS message—submission—time

4) The TLMAU will maintain a one—to—one correlation between MTS message—submission—identifiers and correlation information values to facilitate status query.

10.2.2 *MessageProbe*

The MessageProbe operation is provided by the TLMAU via the Probe—, ProbeAck— and Exception—TAPDUs.

Upon receipt of the Probe—TAPDU by the TLMAU, the TLMAU will take the following actions:

1) The TLMAU will invoke the MTAS import abstract operation ProbeSubmission with the following argument values:

Source of MTAS ProbeSubmission arguments.

See § 10.2.1 — MessageSubmission arguments.

2) If the Probe operation results in an error or if an error is detected in the Probe—TAPDU, the TLMAN will return an Exception—TAPDU to the originator.

3) The TLMAU will, when required, return a ProbeAck—TAPDU to the originator, following the successful completion of the probe operation. The values of the ProbeAck—TAPDU will be set as follows:

Source of ProbeAck—TAPDU components

ProbeAck—TAPDU component		Source
Element name	Value name	
correlationInfo	call—id	CallIdentification that identifies previ Probe—TAPDU being reported on
probeId	probe—msg—id	MTS probe—submission—identifier
submissionTime	submission—time	MTS probe—submission—time

10.2.3 *ExplicitReceive*

The ExplicitReceive operation is provided by the TLMAU via the ExplicitRN—, ExplicitRNAck— and Exception—TAPDUs.

Upon receipt of the ExplicitRN—TAPDU the TLMAU will take the following actions:

1) The TLMAU will invoke the MTAS import abstract operation MessageSubmission with the following argument values:

Source of MTAS MessageSubmission components

MessageSubmission argument	Corresponding ExplicitRN—TAPDU component		Operation
	Element name	Value name	
originator—name	—	—	authenticate user
original—encoded—information—types	—	—	set by TLMAU to “unspecified”
content—type	—	—	set by TLMAU to “IPMS”
content—identifier	—	—	Local matter
priority	priority	priority—ind	
per—message—indicators disclose—recipients conversion—prohibited alternate—recipient—allowed content—return—request	 — — — —	 — — — —	 set to “disclosure—of—recipient—prohibited” set to “conversion—prohibited” set to “alternate—recipient—prohibited” set to “content—return—not—requested”
recipient—name	recipients	recipient—name	
originator—report—request	—	—	set by TLMAU to “no report”
content	—	—	identified as IPN

Note 1 — The IPN submitted as the content is constructed by the TLMAU. ExplicitRN—TAPDU elements representing IPMS elements are mapped onto the corresponding IPMS APDU elements.

Note 2 — If receipt—time is omitted, the TLMAU extracts the Receipt time from the CSS of the session in which this TAPDU was transferred to. This time may differ from the time of actual receipt of IPM.

Note 3 — Set acknowledgment—mode of IPN to "manual".

2) If the Message—Submission operation results in an error, or if an error is detected in the ExplicitRN—TAPDU, the TLMAU will return an Exception—TAPDU to the originator.

3) The TLMAU will, when required, return a ExplicitRNAck—TAPDU to the originator, following the successful completion of the MessageSubmission operation. The values of the ExplicitRNAck—TAPDU will be set as follows:

Source of ExplicitRNAck—TAPDU components

ExplicitRNAck—TAPDU component		Source
Element name	Value name	
correlationInfo	call—id	CallIdentification that identifies previ ExplicitRN—TAPDU being reported
submissionId	submission—msg—id	MTS message—submission—identifi
submissionTime	submission—time	MTS message—submission—time

10.2.4 *MessageCancel*

The MessageCancel operation is provided by the TLMAU via the Cancel— and Exception—TAPDUs.

Upon receipt of the Cancel—TAPDU by the TLMAU, the TLMAU will take the following actions:

The TLMAU will invoke the MTAS abstract operation CancelDeferredDelivery with the following argument value:

Source of CancelDeferredDelivery arguments

CancelDeferredDelivery argument	Corresponding Cancel—TAPDU component		Operation
	Element name	Value name	
message—submission—identifier	submissionId	submission—msg—id	

If the CancelDeferredDelivery operation results in an error, or if an error is detected in the Cancel—TAPDU, the TLMAU will return an Exception—TAPDU to the originating TLM terminal.

10.2.5 *MessageDeliver*

The MessageDeliver operation is provided by the TLMAU via the Deliver—TAPDU.

When the MTAS abstract operation MessageDelivery is invoked by the MTS with an IPM as the MTS message content, the TLMAU will take the following actions:

1) The TLMAU will construct a Deliver—TAPDU for transmission to the destination TLM terminal with the following element values:

Source of Deliver—TAPDU component

Deliver—TAPDU component		Corresponding MessageDelivery argument	Operation
Element name	Value name		
quantityOfDocs	number—of—docs	—	when control Information is conveyed by a normal document, set number of associated documents in Deliver—TAPDU
priority	priority—ind	priority	
originator	originator—name	originator—name	
thisRecipient	this—recipient—name	this—recipient—name	
intendedRecipient	intended—recipient—name	originally—intended—recipient—name	
otherRecipients	other—recipient—name	other—recipient—names	
submissionTime	submission—time	message—submission—time	
timeOfDelivery	delivery—time	message—delivery—time	
deliveryId	delivery—msg—id	message—delivery—identifier	
conversionIndication	eIT	original—encoded—information—types	
conversionIndication	conversion—prohibited	delivery—flags	
convertedInfoTypes	eIT	converted—encoded—information—types	

Note 1 — The IPM received by TLMAU is used to construct the Deliver—TAPDU, Deliver—TAPDU elements, representing MTS and IPMS elements of service, are constructed by the TLMAU from the MessageDeliver operation arguments and IPMS application protocol data unit (APDU) values as indicated above.

Note 2 — Multi—part body message will be sent to the destination TLM terminal by the TLMAU as a multi—document message, each document corresponding to an IP message body part.

2) If the TLMAU is unable to deliver the constructed Deliver—TAPDU to the designation TLM terminal, then an IPN will be constructed for return to the IPMS originator. This IPN will be submitted according to § 10.2.6.

3) The definition of the export port MessageDeliver abstract operation should include a result argument indicating successful delivery or non—delivery. The MTS would then return delivery notifications to the originators of messages routed through a TLMAU only after the result value was indicated.

10.2.6 *ReceiptStatus Notice*

The ReceiptStatusNotice operation is provided by the TLMAU via the ReceiptStatusNotice—TAPDU.

When the MTAS abstract operation MessageDelivery is invoked by the MTS with an IPN as the IPMS content, the TLMAU will take the following actions:

1) The TLMAU will construct a ReceiptStatusNotice—TAPDU for transmission to the destination TLM terminal with the following element values:

Source of ReceiptStatusNotice—TAPDU components

ReceiptStatusNotice—TAPDU component		Corresponding MessageDelivery and receive RN/NRN argument	Operation
Element name	Value name		
quantityOfDocs	number—of—docs	—	when control Information is conveyed by a normal document, set number of associated documents in ReceiptStatusNotice—TAPDU
priority	priority—ind	priority	
deliveryId	delivery—id	message—delivery—identifier	
originator	originator—name	originator—identifier	if this element is omitted, this argument should be constructed from TID obtains from CSS
thisRecipient	this—recipient—name	this—recipient—identifier	
submissionTime	submission—time	message—submission—time	
timeOfDelivery	delivery—time	message—delivery—time	
conversionIndication	eIT	original—encoded—information—types	
conversionIndication	conversion—prohibited	delivery—flags	
convertedInfoTypes	eIT	converted—encoded—information—types	
notificationType	report—type	choice	
subjectIPM	subject—ipm—id	subject—ipm	
iPNOriginator	ipn—originating—user	ipn—originator	
preferredRecipient	preferred—recipient	ipm—preferred—recipient	
timeOfReceipt	receipt—time	receipt—time	
typeOfReceipt	type—of—receipt	acknowledgment—mode	
supplReceiptInfo	suppl—receipt—info	suppl—receipt—info	
nonReceiptInfo	non—receipt—reason	non—receipt—reason	
nonReceiptInfo	discard—reason	discard—reason	
comment	comments	auto—forward—comment	
messageReturnedInd	—	returned—ipm	

Note — What is received by the TLMAU is used to construct the ReceiptStatusNotice—TAPDU. ReceiptStatusNotice—TAPDU elements representing MTS and IPMS elements of service, are constructed by the TLMAU from the MessageDeliver operation arguments and IPMS APDU values, as indicated above.

10.2.7 *DeliveryStatusNotice*

The DeliveryStatusNotice operations are provided by the TLMAU via the DeliveryStatusNotice—TAPDU.

When the MTS abstract operation ReportDelivery is invoked by the MTS, the TLMAU will take the following actions:

1) The TLMAU will construct a DeliveryStatusNotice—TAPDU for transmission to the destination TLM terminal with the following element values:

Source of DeliveryStatusNotice—TAPDU components

DeliveryStatusNotice—TAPDU component		Corresponding ReportDelivery argument	Operation
Element name	Value name		
quantityOfDocs	number—of—docs	—	when control Information is conveyed by a normal document, set number of associated documents in DeliveryStatusNotice—TAPDU
correlationInfo	call—id	—	CallIdentification that identifies previous Send—TAPDU being reported on
priority	priority—ind	priority	
submissionId	submission—id	subject—identifier	
probeId	submission—id	subject—identifier	
reportedRecipient	reported—recipient—name	actual—recipient—identifier	
notificationType	report—type	report	
intendedRecipient	intended—recipient—name	originally—intended—recipient	
convertedInfoTypes	eIT	converted—encoded—information—types	
timeOfDelivery	delivery—time	message—delivery—time	
typeOfUA	type—of—ua	type—of—MTS—user	
nonDeliveryReason	reason—code	non—delivery—reason—code	
nonDeliveryReason	diagnostic—code	non—delivery—diagnostic—code	
contentReturned		returned—content	

2) When required, the TLMAU will accumulate notifications pertaining to a single Send—TAPDU and construct a single DeliveryStatusNotice—TAPDU from multiple ReportDelivery operations.

10.2.8 *Register*

The register operation is provided by the TLMAU via the register—, RegisterAck— and Exception—TAPDUs.

Upon receipt of the Register—TAPDU, the TLMAU will take the following actions:

— If a message delete mode was selected, the TLMAU will subsequently operate according to the new mode with respect to messages output from the DS of the TLM terminal originating the Register—TAPDU.

— If an error recovery mode was selected, the TLMAU will subsequently handle error recovery according to the selected criteria for all transactions with the originator of the Register—TAPDU.

— If a DS mode was selected, the TLMAU will subsequently either hold for retrieval, or auto output messages in the DS of the originator of the Register—TAPDU according to the DS mode selected in this TAPDU.

— If the auto discard mode was enabled by the Register—TAPDU, then the TLMAU will commence automatic deletion of messages in the DS belonging to the originator of the Register—TAPDU when they are obsoleted by subsequent received IPM's.

— If the auto acknowledgement function was enabled by the Register—TAPDU, then the TLMAU will automatically format and submit receipt notifications for subsequent IP messages directed to the originator of the Register—TAPDU. These notifications will be submitted, either following successful delivery of the IP message to the TLM terminal, or upon deposit of the IP message in the TLM terminal's DS.

— If an error is detected with the Register—TAPDU, the TLMAU will return an Exception—TAPDU to the originator.

10.2.9 *DSList*

The DSList operation is implemented by the TLMAU as an internal operation and does not involve the MTS. The DS list operation is provided via the DSQuery—, DSReport— and Exception—TAPDUs as follows:

Upon receipt of the DSQuery—TAPDU by the TLMAU, the TLMAU will take the following actions:

— The TLMAU will prepare a DSReport—TAPDU for return to the originator. If there are no messages in DS, the DSReport—TAPDU will indicate this.

— If an error is detected with the DSQuery—TAPDU, the TLMAU will return an Exception—TAPDU to the originator.

10.2.10 *DSDelete*

The DSDelete operation is implemented by the TLMAU as an internal operation and does not involve the MTS. The DS Delete operation is provided via the DSDelete— and Exception—TAPDUs as follows:

— The TLMAU will delete the indicated message(s) from the DS.

— If an error is detected with the DSDelete—TAPDU or the message indicated is not available for deletion, the TLMAU will return an Exception—TAPDU to the originator.

10.2.11 *DSFetch*

The DSFetch operation is implemented by the TLMAU as an internal operation and does not involve the MTS. The DSFetch operation is provided via the OutputRequest—, OutputMessage— and Exception—TAPDUs as follows:

Upon receipt of the OutputRequest—TAPDU by the TLMAU, the TLMAU will take the following actions:

— For each message indicated in the OutputRequest—TAPDU and found in the DS, the TLMAU will prepare and return an OutputMessage—TAPDU.

— If the delete—after—output function was indicated in the OutputMessage—TAPDU the TLMAU will delete the indicated message(s) from the DS after output.

— If the “auto delete” message delete mode is subscribed to then the TLMAU will delete the indicated message(s) from the DS after output regardless of whether the delete—after—output function was selected in the OutputRequest—TAPDU.

— If an error is detected with the OutputRequest—TAPDU or the message(s) indicated were not available for output, the TLMAU will return an Exception—TAPDU to the originator. If some of a list of indicated messages are available in DS then the TLMAU will output those available, and then return an Exception—TAPDU for those not available or incorrectly indicated.

— If the “auto output” DS mode is subscribed to then the output and associated deletion functions will be executed when the user subscribed conditions are met.

10.2.12 *OutputMessage*

The delivery—time in this TAPDU is the time when the DS received the message.

10.2.13 *MessageStarus*

The MessageStatus operation is implemented by the TLMAU as an internal operation and does not involve the MTS. This operation is applicable only when the TLMAU accumulates notifications for previously submitted multi—address messages. The operations provided by the TLMAU via the StatusQuery—, StatusReport—and Exception—TAPDUs.

Upon receipt of the StatusQuery—TAPDU by the TLMAU, the TLMAU will take the following actions:

— The TLMAU will construct a StatusReport—TAPDU from accumulated notifications pertaining to the message identified in the StatusRequest—TAPDU.

— The TLMAU will not allow StatusQuery operation for ReceiptStatusNotice.

— If an error is detected with the StatusReport—TAPDU or there is no record of the message indicated, the TLMAU will return an Exception—TAPDU to the originator.

11 Formats and encoding of TAPDU

11.1 *Principles*

Elements of a telematic access protocol data unit (TAPDU) shall be coded using human—readable graphic characters of Recommendation T.61 coding scheme. Other coding rules such as mechine—readable coding are for further study.

11.2 *Structure of TAPDU*

1) A TAPDU is composed of one or more documents. The first one contains control information optionally followed by one or more documents with text (message body information). Within one session one or more TAPDU may be conveyed.

2) Control information is conveyed in either a control document or a normal document.

3) The control information is subdivided into a TAPDUs and elements each containing a number field and/or name field, and optionally one or more element value fields. An element number field, which is language independent, and the element name, which is language dependent, uniquely identify an element. In case of international access, the element number field must always be present.

4) The value fields of an element may contain the same TAPDU information types or different TAPDU information types. The element value fields (called components) are categorized as follows:

   — components with pre—defined values, i.e. components with a specific, enumerable set of known, unique values (predefined value);

   — components with a wide range of values which are not pre—defined (general value).

5) There are two different types of component fields:

   — primitive component;

   — constructor component.

6) Each primitive component contains only one parameter. Each constructor component contains more than one parameter.

7) A parameter contains a parameter value, optionally preceded by a Parameter—Id, which identifies the parameters.

8) The formal description of the structure of a TAPDU is shown in Table 3/T.330.

9) A line may contain an Element—Id field and component fields, or the first component field of the element starts on a new line.

10) If the number of characters of the component exceeds the remaining number of characters on this line, the component must be divided into two or more lines by “NL” function. However, it is not allowed to divide the component within a parameter.

TABLE 3/T.330

The structure of TAPDUs

```
        TAPDU ::= SEQUENCE {
                                        ControlInfo,
                                        SEQUENCE OF MessageBodyInfo OPTIONAL },
ControlInfo ::= SEQUENCE {
                                        TAPDUId,
                                        Elements OPTIONAL }
TAPDUId ::= SEQUENCE {
                                        TAPDUNumber OPTIONAL,
                                        TAPDUName OPTIONAL }
    —— One of this must be present.
Elements ::= SEQUENCE {
                                        ElementId,
                                        ElementValues }
ElementId ::= SEQUENCE {
                                        ElementNumber OPTIONAL,
                                        ElementName OPTIONAL }
    —— One of this must be present.
ElementValues ::= SET OF Component
    —— See Note
Component ::= CHOICE {
                                        PrimitiveComponent,
                                        ConstructorComponent }
PrimitiveComponent ::= Parameter
ConstructorComponent ::= SET OF Parameter
Parameter ::= SEQUENCE {
                                        ParameterId OPTIONAL,
                                        ParameterValue }
```

Note — Order of components as prescribed by the TAPDU descriptions in § 10.

11.3 *Coding rule*

11.3.1 *TAPDU ID*

1) The TAPDU number assigned to TAPDU shall consist of two parts separated by a “period” (.). The first part identifies the application, for example, “3” is assigned to this application. The second part identifies the procedures specified in the application.

2) Where national requirements dictate the use of non—standardized TAPDU numbers. Administrations may choose any values in the range 1000—1999 for the first part of non—standardized application identifiers.

3) Other rules applied to TAPDU number and name are same as those of the element number and name, described below.

11.3.2 *Element ID*

1) The element number shall be sequentially assigned a different number.

2) An element number is always closed by the character “colon” (:).

3) There shall be no restriction of the number of digits for element numbers and any leading zeros are ignored.

4) Where national requirements dictate the use of non—standardized element numbers Administrations may choose any values in the range 1000—1999 for non—standardized elements.

5) The element number and the element name shall be separated by the character “space”.

6) An element name shall be represented by a text string, that is a sequence of graphic characters. Capital and small characters have the same effect.

7) An element name is always closed by the character “colon” (:).

11.3.3 *Element value fields*

For unregistered TLM—users with international access, the pre—defined values as defined in the following tables have to be applied. For all other cases, these values can be replaced by nationally defined values.

11.3.4 *Separators and common rules*

1) TAPDU—Ids and elements shall be preceded by the following delimiters:
   — “CR LF” sequence, or
   — “CR LF BS +” sequence.
2) The Element—Id and the first component shall be separated by the character “space” or “New Line” functions (“NL” = “CR LF” or “LF CR”).
3) Components shall be separated by “comma” (“,”) and optionally “NL”.
4) When components with pre—defined and not pre—defined values are contained in an element, they shall be separated by a “NL” and the line with the pre—defined values should start with the character “=”.
5) Parameters within one component field shall be separated by the character “slash” (“/”) or “semicolon” (“;”). “CR LF” within a parameter is not allowed, except if the parameter is longer than 1 line.
6) The actual value of a parameter value is encoded by a sequence of graphic characters. Capital and small characters have the same effect.
7) If some pre—defined values are absent but required, then their default value shall apply.
8) The element ID and the first element value field shall be separated by the character “space” or the “NL” function.
9) Contiguous “NL” and “LF” are considered as one “NL”.
10) Contiguous embeded space are considered as one space. Leading spaces in a line are ignored.
11) The character sequence “Space //” indicates that the following of the line shall be considered as a comment.

11.4 *Format of TAPDU*

The format of each TAPDU according to the above coding rules is shown in Annex C of this Recommendation.

11.5 *Reference between TAPDU components and its coding format*

This section provides the tables necessary for the encoding of TAPDU components.

11.5.1 *TAPDUId and elementId* (see Table 4/T.330)

Table 4/T.330 comprises four columns:

1) The first column contains the TAPDUId or Element—Id name as used in the ASN.1 description of § 10.
2) The second column contains the type of this element:
   a) primitive: the element contains only one elementValue field;
   b) constructor: the element may contain more than one elementValue field.
3) The third column contains the actual coding format of the TAPDUId or element—Id.
4) The last column contains remarks.

11.5.2 *ElementValues* (see Table 5/T.330)

Table 5/T.330 comprises five columns:

1) The first column contains the ElementValue name (component name) as used in the ASN.1 description of § 10.
2) The second column contains the type of ElementValue field:
   a) primitive: the component contains only one parameter;

b) constructor: the component may contain more than one parameter.

3) The third column contains the type of the value:

a) predefined;

b) general, as defined in this section.

4) The fourth column contains the actual coding format, or, in case of general value, a reference name which points to the actual coding format in Table 6/T.330.

5) The last column contains remarks.

11.5.3 *General values* (see Table 6/T.330)

Table 6/T.330 comprises five columns:

1) The first column contains the reference name (general value name) used in Table 5/T.330.

2) The second column contains the name of the parameter.

3) The third column contains the code of the value.

4) The fourth column contains the keyword and format of this parameter.

5) The last column contains remarks.

TABLE 4/T.330

Format encoding of TAPDU and ElementId

TAPDUId and ElementId name	Type	T.61 Character coding format	Remarks
authorizing	Constructor	21: AUTHORIZING:	
autoFWDComment	Primitive	79: AUTO—FWD—COMMENT:	
autoFWDHeading		78: AUTO—FWD—HEADING:	for further study
autoFWDIPMs	Primitive	76: AUTO—FWD—IPMS:	
autoFWDRecipients	Constructor	77: AUTO—FWD—RECIPIENTS:	
autoOutput	Constructor	60: AUTO—OUTPUT:	
bcc	Constructor	24: BCC:	
bodyType	Constructor	31: BODY—TYPE:	
cancel	—	3.13: CANCEL:	
cc	Constructor	23: CC:	
comment	Primitive	50: COMMENT:	
contentIndicator	Constructor	18: CONTENT—INDICATOR:	
contentInfo	Primitive	17: CONTENT—INFO:	
contentReturned	—	72: CONTENT—RETURNED—INDICATION:	
conversion	Primitive	16: CONVERSION:	
conversionIndication	Constructor	42: CONVERSION—INDICATION:	
convertedInfoTypes	Primitive	44: CONVERTED—INFORMATION—TYPES:	
correlationInfo	Primitive	1: CORRELATION—INFORMATION:	
deleteAfterOutput	Primitive	80: DELETE—AFTER—OUTPUT:	
deliver	—	3.3: DELIVER:	
deliveryId	Primitive	35: DELIVERY—ID:	
deliveryStatusNotice	—	3.4: DELIVERY—STATUS—NOTICE:	
dsMode	Primitive	58: DS—MODE:	
dsQuery	—	3.7: DS—QUERY:	
dsReport	—	3.8: DS—REPORT:	
errors	Primitive	9: ERRORS:	
exception	—	3.12: EXCEPTION:	
expiredDiscard	Primitive	73: EXPIRED—DISCARD:	
explicitRN	—	3.6: EXPLICIT—RN:	
explicitRNAck	—	3.16: EXPLICIT—RN—ACK:	
forwardedInfo	Constructor	32: FORWARDED—INFO:	
from	Primitive	20: FROM:	

TABLE 4/T.330 *(cont.)*

TAPDUId and ElementId name	Type	T.61 Character coding format	Remarks
orgIntendedRecipient	Primitive	40: INTENDED—RECIPIENT:	
iPNOriginator	Primitive	69: IPN—ORIGINATOR:	
language	Primitive	53: LANGUAGE:	
latestDelivery	Primitive	34: LATEST—DELIVERY:	
messageDelete	—	3.18: MESSAGE—DELETE:	
messageDeleteMode	Primitive	81: MESSAGE—DELETE—MODE:	
messageLength	Primitive	37: MESSAGE—LENGTH:	
messageReturnedInd	—	51: MESSAGE—RETURNED—INDICATION:	
messageSelector	Primitive	82: MESSAGE—SELECTOR:	
messageStatus	Primitive	83: MESSAGE—STATUS:	
messageType	Primitive	52: MESSAGE—TYPE:	
msgIncomplete	—	67: MSG—INCOMPLETE:	This element has not value
nonDeliveryReason	Primitive	46: NON—DELIVERY—REASON:	
nonReceiptInfo	Primitive	49: NON—RECEIPT—INFO:	
		43: NOTIFICATION—TYPE:	
obsoletedDiscard	Primitive	74: OBSOLETED—DISCARD:	
obsoletedIPMs	Constructor	29: OBSOLETED:	
otherRecipients	Constructor	41: OTHER—RECIPIENTS:	
outputMessage	—	3.10: OUTPUT—MESSAGE:	
outputRequest	—	3.9: OUTPUT—REQUEST:	
perMessageIndicators	Constructor	19: FLAGS:	
preferredRecipient	Primitive	70: PREFERRED—RECIPIENT:	
priority	Primitive	13: PRIORITY:	
probe	—	3.2: PROBE:	
probeAck	—	3.15: PROBE—ACK:	
probeId	Primitive	66: PROBE—ID:	
quantityOfDocs	Primitive	62: QUANTITY—OF—DOCS:	
recipients	Constructeur	15: RECIPIENTS:	
receiptStatusNotice	—	3.5: RECEIPT—STATUS—NOTICE:	
redirectedFrom	Constructeur	54: REDIRECTED—FROM:	
register	—	3.11: REGISTER:	
registerAck	—	3.17: REGISTER—ACK:	

TABLE 4/T.330 *(end)*

TAPDUId and ElementId name	Type	T.61 Character coding format	Remarks
relatedIPMs	Constructor	28: RELATED—IPMS:	
repliedToIPM	Primitive	30: REPLIED—TO—IPM:	
reply	Constructor	25: REPLY:	
reportedMessageId	Primitive	75: REPORTED—MESSAGE—ID:	
reportedRecipient	Primitive	3: REPORTED—RECIPIENT:	
retrievalId	Primitive	38: RETRIEVAL—ID:	
returnAddress	Primitive	36: RETURN—ADDRESS:	
send	—	3.1: SEND:	
sendAck	—	3.14: SEND—ACK:	
statusQuery	—	3.19: STATUS—QUERY:	
statusReport	—	3.20: STATUS—REPORT:	
subject	Primitive	26: SUBJECT:	
subjectIPM	Primitive	71: SUBJECT—IPM:	
submissionId	Primitive	65: SUBMISSION—ID:	
submissionTime	Primitive	33: SUBMISSION—TIME:	
supplInfo	Primitive	68: SUPPLEMENTARY—INFORMATION:	
supplReceiptInfo	Primitive	68: SUPPLEMENTARY—INFORMATION:	
thisIPM	Primitive	27: THIS—IPM:	
thisRecipient	Primitive	39: THIS—RECIPIENT:	
timeOfDelivery	Primitive	4: TIME—OF—DELIVERY:	
timeOfReceipt	Primitive	47: TIME—OF—RECEIPT:	
timeOfReport	Primitive	84: TIME—OF—REPORT:	
tLMAUOperation	Constructor	59: TLMAU—OPERATION:	
to	Constructor	22: TO:	
typeOfReceipt	Primitive	48: TYPE—OF—RECEIPT:	
typeOfUA	Primitive	45: TYPE—OF—UA:	

Conventions:

1) Primitive: element contains only one element value field.

2) Constructor: element may contain more than one element value field.

3) abcde: underlined characters, i.e. “abc” are mandatory in case of international access (see § 11.2).

4) : space character.

TABLE 5/T.330

Format encoding of elements values

Element value name	Type of element value field	Type of value	T.61 character coding format	Remarks
alternate—recipient—allowed	Primitive	Predefined	Allowed	
authorizing	Constructor	—	21: AUTHORIZING:	
authorizing—user	Constructor	General	*R Descriptor*	
auto—acknowledgment	Primitive	Predefined	Auto—Receipt, Manual—Receipt (défaut)	
auto—forwarded	Primitive	Predefined	Auto—forwarded,	
Not—Auto—forwarded (default)				
auto—fwd—Comment	Primitive	General	*any Text*	
auto—fwd—ipms	Primitive	Predefined	Auto—forwarded,	
Not—Auto—forwarded (default)				
auto—fwd—recipient—name	Constructor	General	*OR Name*	
blind—copy—recipient	Constructor	General	*OR Descriptor*	
body—part	Primitive	Predefined	IA5Text, TLX, Voice, G3Fax, G4Fax—Class1, TTX, Videotex, Message, Mixed—Mode, Encrypted	
call—id	Primitive	General	*Call Identification*	
comments	Primitive	General	*Comments*	
content—return—request	Primitive	Predefined	Content—Return—Request	
conversion—info	NO, Yes; WLOSS			
copy—recipient	Constructor	General	*OR Descriptor*	
deferred—delivery—time	Primitive	General	*Date and Time*	
delete—after—output	Primitive	Predefined	Keep, Delete	
delivery—msg—id	Primitive	General	*Message Identifier*	
	Primitive	General	*Date and Time*	
diagnostic—code [a)]	Primitive	Predefined	Unrecognized—OR—Name, Ambiguous—OR—Name, MTS—Congestion, Loop—Detected, Recipient—Unavailable, Maximum—Time—Expired, Content—Too—Long, Conversion—Impractical, Encoded—Information—Type—Unsupported, Conversion—Prohibited, Invalid—Arguments, Implicit—Conversion—Not—Subscribed, Content—Syntax—Error, Pragmatic—Constraint—Violation, Protocol—Violation, Content—Not—Supported, Too—Many—Recipient, No—Bilateral—Agreement	

TABLE 5/T.330 *(cont.)*

Element value name	Type of element value field	Type of value	T.61 character coding format	Remarks
discard—ipm	Primitive	Predefined	Discard (default), Not—Discard	
discard—reason	Primitive	Predefined	IPM—Expired, IPM—Obsoleted, User—Subscription—Terminated	
disclose—recipients	Primitive	Predefined	No—Disclosure	
dsMode	Primitive	—	58:DS—MODE:	
ds—mode	Primitive	Predefined	Auto—Output, Retrieval	
eIT	Primitive	Predefined	IA5Text, TLX, Voice, G3Fax, G4Fax—Class1, TTX, Videotex, Undefined, Mixed—Mode	
error—cause	Primitive	Predefined	IPMS—Element—of—Service—Not—Subscribed,* MTS—Element—of—Service—Not—Subscribed,* Name—Malformed,* IPM—Not—Submitted, IPM—Transferred, IPM—Delivered, Element—of—Service—Not—Subscribed,* Message—Delivered, Message—Transferred, Originator—Invalid,* Query—Identifier—Invalid,* Recipient—Improperly—Specified,* Submission—Identifier—Invalid,* No—Message—in—DS, DS—Not—Supported, DS—Not—Subscribed, Retrieval—Identifier—Invalid,* Parameter—Invalid,* Not—Changed	* optionally followed by the name, service, parameter, etc. concerned in « . . . »
error—recovery—mode	Primitive	Predefined	Recovery—1, Recovery—2, Recovery—3	
expiry—time	Primitive	General	*Date and Time*	
explicit—conversion	Primitive	Predefined	TLX, IA5, G3, G4, VTX, TTX	
forwarded—time	Primitive	General	*Date and Time*	
frequency	Primitive	General	*Frequency*	
importance	Primitive	Predefined	Low, Normal (default), High	
intended—recipient—name	Constructor	General	*OR Name*	
ipn—originating—user	Constructor	General	*OR Descriptor*	
language—ind	Primitive	Predefined		ffs
latest—delivery—time	Primitive	General	*Date and Time*	
message—delete—mode	Primitive	Predefined	Auto—Delete (default), Manual—Delete	
message—length	Primitive	General	*Message Length*	
messageType	Primitive	—	52:MESSAGE—TYPE:	
non—receipt—reason	Primitive	Predefined	IPM—DISCARD, IPM—Auto—forwarded	
nrn—request	Primitive	Predefined	NRN—Request	
number—of—docs	Primitive	General	*Number Of Associated Documents*	

TABLE 5/T.330 *(cont.)*

Element value name	Type of element value field	Type of value	T.61 character coding format	Remarks
obsoleted—ipm—id	Constructor	General	*IPM Identifier*	
originating—user	Constructor	General	*OR Descriptor*	
originator—name	Constructor	General	*OR Name*	
originator—requested—alternate—recipient	Constructor	General	*OR Name*	
other—recipient—name	Constructor	General	*OR Name*	
output—time	Primitive	General	*Date and Time*	
Physical—delivery—mode	Primitive	Predefined	"PDM="OM (defaut), EMS, SPEC, COL, TLXA, TTXA, PHA, BFAX	
Physical—delivery—report— request	Primitive	Predefined	"REP="UND (defaut), PDS, MHS, PDMHS	
Physical—forwarding—address—request	Primitive	Predefined	PFAR	
Physical—forwarding—prohibited	Primitive	Predefined	PFP	
postal—address	Primitive	General	*OR Name*	
preferred—recipient	Constructor	General	*OR Descriptor*	
primary—recipient	Constructor	General	*OR Descriptor*	
priority—ind	Primitive	Predefined	Urgent, Non—Urgent, Normal (défaut)	
probe—msg—id	Primitive	General	*Message Identifier*	
reason—code	Primitive	Predefined	Transfer—Failure, Unable—To—Transfer, Conversion—Not—Performed	
receipt—time	Primitive	General	*Date and Time*	
recipient—name	Constructor	General	*OR Name*	
recipient—number—for—advice	Primitive	General	*CALL="Number*	
recipient—reassignment—prohibited	Primitive	Predefined	RRP	
redirected—from	Constructor	General	*OR Name*	
registered—mail—type	Primitive	Predefined	NRM (defaut), RM, RMA	
related—ipm—id	Constructor	General	*IPM Identifier*	
replied—to—ipm—id	Constructor	General	*IPM Identifier*	
reply—recipient	Constructor	General	*OR Descriptor*	
reply—request	Primitive	Predefined	Reply, No—Reply (default)	
reply—time	Primitive	General	*Date and Time*	
reported—message—id	Primitive	General	*Message Identifier*	

TABLE 5/T.330 *(end)*

Element value name	Type of element value field	Type of value	T.61 character coding format	Remarks
reported—recipient—name	Constructor	General	*OR Name*	
report—time	Primitive	General	*Date and Time*	
report—type	Primitive	Predefined	Receipt, Non—Receipt, Delivery, Non—delivery	
requested—delivery—method	Primitive	Predefined	"RDL="ANY (défaut), MAS, PD, TLX, TTX, G3, G4, IA5, VTX	
retrieval—id	Primitive	General	*Retrieval Identifier*	
return—request	Primitive	Predefined	Return—Request	
rn—request	Primitive	Predefined	RN—Request	
sensitivity	Primitive	Predefined	Personal, Private, Company—Confidential	
status	Primitive	Predefined	In—Process, Delivered, Non—Delivered	
subject—content	Primitive	General	*Subject*	
subject—ipm—id	Primitive	General	*Message Identifier*	
submission—msg—id	Primitive	General	*Message Identifier*	
submissionTime	Primitive	—	33:SUBMISSION—TIME:	
suppl—info	Primitive	General	*Supplementary Information*	
suppl—receipt—info	Primitive	General	*Supplementary Information*	
terminal—type	Primitive	Predefined	"TTyp=" TLX, TTX, G3, G4, IA5, vtx	
this—recipient—name	Constructor	General	*OR Name*	
this—ipm—id	Constructor	General	*IPM Identifier*	
type—of—receipt	Primitive	Predefined	Manual (défaut), Automatic	
type—of—ua	Primitive	Predefined	Private, Public (défaut)	
user—report—request	Primitive	Predefined	No—Report, Non—Delivery—Report, Report	

[a] More diagnostic—codes can be found in Recommendation X.411 and should be translated into appropriate T.61 text.

Note — The character strings in italics in the fourth column are the entry name of the General value list, Table 6/T.330.

Conventions:

1) Primitive: elementary or value contains only one component.

2) Constructor: element or value contains more than one component.

3) abcde: underlined characters, i.e. "abc" are mandatory.