MONTAGE: FIN DU § 9 EN T | TE DE CETTE PAGE

How a TLMAU realizes the mhs-doc-xfer port by means of which it interacts with a TLM terminal is specified in this section. But how a TLMA realizes the ports by means of which it interacts with a TLM user and MTS is outside the scope of this Recommendation.

Telematic access protocol for accessing to IPMS, called P5 protocol, is provided to realize the interaction, which means abstract operations performed at the mhs-doc-xfer port, between a TLMAU and a TLM terminal. The concrete interactions, which correspond to abstract operations, are realized as telematic access protocol data units (TAPDUs).

It should be noted that the TLMAU may not support all the conditional TAPDUs and all the optional elements or parameters of a TAPDU. The actual support of the TAPDUs and parameters depends on the application and the version of the colocated MTA.

The relationship between abstract operations at the mhs-doc-xfer port and associated TAPDUs are summarized in Table 2/T.330.

10.1 *Description of TAPDU*

10.1.1 *MessageSend*

10 Realization of abstract operations

A Send-TAPDU to invoke the MessageSend abstract operation. The TLMAU returns a SendAck-TAPDU to report the result of that operation, or may return an Exception-TAPDU (§ 10.1.1.3) to report an abstract error.

H.T. [T20.330]

TABLE 2/T.330

Relationship between abstract operation and TAPDU

{ mhs-doc-xfer Abstract operation } Direction of transfer TLM TLMAU } Operation	TAPDU TAPDU name	{ TLMAU status	
MessageSend	(O) Send-TAPDU (R) SendAck-TAPDU (E) Exception-TAPDU	M C M	| (ra ← ←
MessageProbe	(O) Probe-TAPDU (R) ProbeAck-TAPDU (E) Exception-TAPDU	C C C	| (ra ← ←
ExplicitReceive	(O) ExplicitRN-TAPDU (R) ExplicitRNAck-TAPDU (E) Exception-TAPDU	C C C	| (ra ← ←
MessageCancel	(O) Cancel-TAPDU (R) — (E) Exception-TAPDU	C — C	| (ra ←
MessageDeliver	(O) Deliver-TAPDU	M	←
ReceiptStatusNotice (O) ReceiptStatusNotice-TAPDU }	{ M	 ←	
DeliveryStatusNotice (O) DeliveryStatusNotice-TAPDU }	{ M	 ←	
Register	(O) Register-TAPDU (R) RegisterAck-TAPDU (E) Exception-TAPDU	C C C	| (ra ← ←
DSList	(O) DSQuery-TAPDU (R) DSReport-TAPDU (E) Exception-TAPDU	C C C	| (ra ← ←
DSDelete	(O) MessageDelete-TAPDU (R) — (E) Exception-TAPDU	C — C	| (ra ←
DSFetch	(O) OutputRequest-TAPDU (R) OutputMessage-TAPDU (E) Exception-TAPDU	C (remarque 1) C (remarque 1) C (remarque 1)	| (ra ← ←
MessageStatus ← O Argument R Result E Error M Mandatory C Conditional }	(O) StatusQuery-TAPDU (R) StatusReport-TAPDU (E) Exception-TAPDU	C C C	| (ra ← {

Note 1 — In cases where TLMAU provides DS, these TAPDU are mandatory.

Note 2 — A message may arrive at a TLM terminal as a result of either a Deliver-TAPDU or OutputMessage-TAPDU. The Deliver-TAPDU is applicable when delivery occurs directly to a TLM erminal. The OutputMessage-TAPDU is only applicable in the case that DS is subscribed.

10.1.1.1 *Send-TAPDU*

The Send-TAPDU comprises following elements:

H.T. [1T21.330]

{
Send-TAPDU
Send-TAPDU ::= SEQUENCE { |

Send [0]
SEQUENCE { |

Send ::= send [0]
SendTAPDUId,

Send ::=
send [1]
SEQUENCE { |

Send ::= S [0] quanti-
tyOfDocs QuantityOfDocsElementId,

Send ::= S [0]
number-of-docs NumberOfAssociatedDo-
cuments } | PTIONAL }

-- See Note 1

Send [1]
SET { |

Send ::=
send [0]
SEQUENCE { |

Send ::= S [0] prior-
ity PriorityElementId,

Send ::= S [0]
priority-ind PriorityValue DAFAULT
normal } | PTIONAL,

Send ::=
send [1]
SEQUENCE { |

Send ::= S [0] perMes-
sageIndicators PerMessageIndicatorsElementId,

Send ::= **send [1]**
SEQUENCE { | fR **SEQUENCE {**
|

Send ::= S [0] deferred-delivery-time [0]
DateandTime OPTIONAL,

Send ::= **send [1] SEQUENCE {**
| SEQUENCE { | fR **[1] SET { |**

Send ::= S [0]
disclose-recipients [0] DiscloseReci-
pientsValue OPTIONAL,

Send ::= S [0]
alternate-recipient-allowed
[1] AlternateRecipientAl-
lowedValue OPTIONAL,

Send ::= S [0]
recipient-reassignment-prohibited
[2] ReassignmentValue

OPTIONAL } } } OPTIONAL,

Send ::=
send [2]
SEQUENCE { |

Send ::= S [0]
conversion Conver-
sionElementId,

Send ::= S [0]
conversion-info ConversionIn-
foValue } | PTIONAL,

Send ::=
send [3]
SEQUENCE { |

Send ::= S [0] conten-
tinfo ContentInfoElementId,

Send ::= S [0]
content-return-request ContentReturnRequest-
Value } | PTIONAL,

Send ::=
send [4]
SEQUENCE { |

Send ::= S [0] retur-
nAddress ReturnAddressElementId,

Send ::= S [0]
postal-address PostalAd-
dressValue OPTIONAL,

Send ::=
send [5]
SEQUENCE { |

Send ::= S [0] latest-
Delivery LatestDeliveryElementId,

Send ::= S [0]
latest-delivery-time DateandTime } |
PTIONAL }

Send ::=
send [6]
SEQUENCE { |

Send ::= S [0]
to
ToElementId,

Send ::= send [6]
SEQUENCE { | fR SET OF SEQUENCE { |

Send ::= S [0]
primary-recipient [0]
ORDescriptor,

primary-recipient [1] RecOptions }
} | PTIONAL,

-- See Note 2			
Send		**::=**	
send			[7]
SEQUENCE {			
Send	**::=**	**S**	**[0]**
cc			
CCElementId,			
Send	**::=**		**send [7]**
SEQUENCE {	**|**	**fR**	**SET OF SEQUENCE {**
|			
Send	**::=**	**S**	**[0]**
copy-recipient			[0]
ORDescriptor,			
copy-recipient			[1] RecOptions }
} | PTIONAL,			
-- See Note 2			
Send		**::=**	
send			[8]
SEQUENCE {			
Send	**::=**	**S**	**[0]**
bcc			
BCCElementId,			
Send	**::=**		**send [8]**
SEQUENCE {	**|**	**fR**	**SET OF SEQUENCE {**
|			
Send	**::=**	**S**	**[0]**
blind-copy-recipient			[0]
ORDescriptor,			
blind-copy-recipient			[1] RecOptions }
} | PTIONAL }			
-- See Note 2			
			}

Tableau [1T21.330], p.2

H.T. [2T21.330]

{
Send-TAPDU *(continued)*
-- Send-TAPDU Definition (continued)

Send [2]
SET { |

Send ::=
send [0]
SEQUENCE { |

Send ::= S [0]
thisIPM ThisIP-
MElementId,

Send ::= S [0]
this-ipm-id IPMIdentifier } |
PTIONAL,

-- See Note 3

Send ::=
send [1]
SEQUENCE { |

Send ::= S [0]
from
FromElementId,

Send ::= S [0]
originating-user ORDescriptor }
| PTIONAL,

Send ::=
send [2]
SEQUENCE { |

Send ::= S [0] author-
izing AuthorizingElementId,

Send ::= send [8]
SEQUENCE { | fR SET
OF

Send ::= S [0]
authorizing-user OrDescriptor } |
PTIONAL,

Send ::=
send [3]
SEQUENCE { |

Send ::= S [0] replied-
ToIPM RepliedToIPMElementId,

Send ::= S [0]
replied-to-ipm-id IPMIdentifier } |
PTIONAL,

Send ::=
send [4]
SEQUENCE { |

Send ::= S [0]
obsoletedIPMs
ObsoletedIPMsElementId,

Send ::= send [4]
SEQUENCE { | fR SEQUENCE
OF

Send ::= S [0]
obsoleted-ipm-id IPMIdentifier } |
PTIONAL,

Send ::=
send [5]
SEQUENCE { |

Send ::= S [0] rela-
tedIPMs RelatedIPMsElementId,

Send ::= send [4]
SEQUENCE { | fR SEQUENCE
OF

Send ::= S [0]
related-ipm-id IPMIdentifier } |
PTIONAL,

Send ::=
send [6]
SEQUENCE { |

Send ::= S [0] sub-
ject SubjectElementId,

Send ::= S [0]
subject-content SubjectContent }
| PTIONAL,

Send ::=
send [7]
SEQUENCE { |

Send ::= S [0] conten-
tIndicator ContentIndicatorElementId,

Send ::= send [7]
SEQUENCE { | fR SEQUENCE {
|

Send ::= S [0]
expiry-time [0] DateandTime
OPTIONAL,

expiry-time [1]
SET { |

Send ::= S [0] impor-
tance [0] ImportanceValue DEFAULT normal,

Send ::= S [0] sensi-
tivity [1] SensitivityValue OPTIONAL } } } | PTIONAL,

Send ::=
send [8]
SEQUENCE { |

Send ::= S [0]
reply
ReplyElementId,

Send ::= SEQUENCE { **|** **fR** **|**	**send [8] SEQUENCE {**
Send ::= S reply-time [0] DateandTime,	**[0]**
reply-time OF { |	[1] SET
Send ::= S reply-recipient } | PTIONAL,	**[0]** ORDescriptor }
Send **send** SEQUENCE { |	**::=** [9]
Send ::= S language ageElementId,	**[0]** Langu-
Send ::= S language-ind | PTIONAL,	**[0]** LanguagueInd }
MsgIncomplete | 10] MsgIncompleteElementId OPTIONAL }	
-- Body	
Send SET { |	[3]
Send **send** SEQUENCE { |	**::=** [0]
Send ::= S [0] Type	Body- BodyTypeElementId,
Send ::= SEQUENCE { **|** **fR** **|**	**send [0] SET OF {**
Send ::= S Body-part PTIONAL } }	**[0]** } BodyPartValue } |
_	

Tableau [2T21.330], p.3

H.T. [3T21.330]

{
Send-TAPDU *(end)*
-- Send-TAPDU Definition (continued)

--
Definition of RecOptions

RecOptions
::= SET { |

RecOptions ::= SET { | fR
user-report-request
[1]
UserReportRequestValue

user-report-request
[1]

OPTIONAL,

RecOptions ::= SET { | fR
explicit-conversion
[2]
ExplicitConversionValue

user-report-request
[2]

OPTIONAL OPTIONAL,

RecOptions ::= SET { | fR
rn-request
[3]
RNRequestValue OPTIONAL,

RecOptions ::= SET { | fR
nrn-request
[4]
NRNRequestValue OPTIONAL,

RecOptions ::= SET { | fR
return-request
[5]
ReturnRequestValue OPTIONAL,

RecOptions ::= SET { | fR
reply-request
[6]
ReplyRequestValue DEFAULT

user-report-request
[6]

noReply,

RecOptions ::= SET { | fR
requested-delivery-method
[7]
RequestedDelValue
OPTIONAL,

RecOptions ::= SET { | fR
terminal-type
[8]
TerminalTypeValue

OPTIONAL,

RecOptions ::= SET { | fR
physical-forwarding-prohibited
[9]
PhyForProhibValue
OPTIONAL,

RecOptions ::= SET { | fR
physical-forwarding-address-request
[10] PhyForAdrValue
OPTIONAL,

RecOptions ::= SET { | fR
physical-delivery-modes [11] PhyDelModValue OPTIONAL,

RecOptions ::= SET { | fR
registered-mail-type [12] RegMailTypeValue OPTIONAL,

RecOptions ::= SET { | fR
recipient-number-for-advice [13] RecNumAdvValue OPTIONAL,

RecOptions ::= SET { | fR
physical-delivery-report-request
[14] PhyDelRepValue
OPTIONAL,

RecOptions ::= SET { | fR
originator-requested-alternate-recipient
[15]
OrgRecAltValue } OPTIONAL,

{

Note 1 — This element must be present when ControlInfo is conveyed by a normal document and more than one TAPDU are conveyed in this session.

Note 2 — OR Descriptor must contain an ORAddress and at least one of these addresses must be present.

Note 3 — When this 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.

}

Tableau [3T21.330], p.4

10.1.1.2 *SendAck-TAPDU*

The SendAck-TAPDU comprises following elements:

H.T. [T22.330]

{	
SendAck-TAPDU	
SendAck-TAPDU ::= SEQUENCE { |	
Send	[0]
SEQUENCE { |	
Send ::=	sendAck [0]
SendAckTAPDUId,	
Send ::= sendAck	[1]
SEQUENCE { |	
Send ::= S [0]	corre-
lationInfo	CorrelationInfoElemen-
tId,	
Send ::= S [0]	
call-id	
CallIdentification } }	
-- *See Note*	
Send	[1]
SET { |	
Send ::= send	[0]
SEQUENCE { |	
Send ::= S [0]	sub-
missionId	SubmissionIdElemen-
tId,	
Send ::= S [0]	
submission-msg-id	
MessageIdentifier }	
Send ::= send	[1]
SEQUENCE { |	
Send ::= S [0]	sub-
missionTime	SubmissionTimeElemen-
tId,	
Send ::= S [0]	
submission-time	
DateandTime } } }	
}	
{	
Note — This element is a session connection information that identifies previous Send-TAPDU being reported on.	
}	

Tableau [T22.330], p.

10.1.1.3 *Exception-TAPDU*

The Exception-TAPDU comprises following elements:

H.T. [T23.330]

{
Exception-TAPDU
Exception-TAPDU ::= SEQUENCE { |
 	
Send [0]
SEQUENCE { |
 	
Send ::= exception [0]
ExceptionTAPDUId,
 	
Send ::= exception [1]
SEQUENCE { |
 	
Send ::= S [0] corre-
lationInfo CorrelationInfoElemen-
tId,
 	
Send ::= S [0]
call-id
CallIdentification }
 	
-- See Note
 	
Send ::= exception [2]
SEQUENCE { |
 	
Send ::= S [0]
errors Error-
sElementId,
 	
Send ::= S [0]
error-cause Error-
CauseValue } } }
}

{
 	
Note — This element is a session connection information that identifies associated TAPDU being reported on e.g. Send-TAPDU.
 	
}

Tableau [T23.330], p.

10.1.2 *MessageProbe*

A TLM terminal sends a Probe-TAPDU to invoke the MessageProbe abstract operation. The TLMAU returns a ProbeAck-TAPDU to report the result of that operation, or may return an Exception-TAPDU (§ 10.1.1.3) to report an abstract error.

10.1.2.1 *Probe-TAPDU*

The Probe-TAPDU comprises following elements:

H.T. [T24.330]

{	
Probe-TAPDU	
Probe-TAPDU ::= SEQUENCE { |	
Send	[0]
SEQUENCE { |	
Send ::=	probe [0]
ProbeTAPDUId,	
Send ::= probe	[1]
SEQUENCE { |	
Send ::= S [0]	quan-
tityOfDocs	QuantityOfDocsElemen-
tId,	
Send ::= S [0]	
number-of-docs	NumberOfAssociated-
Documents } | PTIONAL }	
Send	[1]
SET { |	
Send [1] SET { | fR	-- *Continuation*
see send-TAPDU	
Send [1] SET { | fR	-- *Note that only few elements*
of the send-TAPDU are relevant for a	
Probe-TAPDU	
Send [1] SET { | fR	-- *Not relevant*
elements will be ignored	
Send [1] SET { | fR	-- *At least one*
recipient must be present	
}	

Tableau [T24.330], p.

10.1.2.2 *ProbeAck-TAPDU*

The ProbeAck-TAPDU comprises following elements:

{ ProbeAck-TAPDU ProbeAck-TAPDU ::= SEQUENCE { | **Send** [0] SEQUENCE { | **Send ::=** probeAck [0] ProbeAckTAPDUId, **Send ::= probeAck** [1] SEQUENCE { | **Send ::= S [0]** correlationInfo CorrelationInfoElementId, **Send ::= S [0]** call-id CallIdentification } } **Send** [1] SET { | **Send ::= send** [0] SEQUENCE { | **Send ::= S [0]** probeId ProbeElementId, **Send ::= S [0]** probe-msg-id MessageIdentifier } **Send ::= send** [1] SEQUENCE { | **Send ::= S [0]** submissionTime SubmissionTimeElementId, **Send ::= S [0]** submission-time } DateandTime } } }

Tableau [T25.330], p.

10.1.3 *ExplicitReceive*

A TLM terminal sends an ExplicitRN-TAPDU to invoke the ExplicitReceive abstract operation. The TLMAU returns an ExplicitRNAck-TAPDU to report the result of that operation, or may return an Exception-TAPDU (see § 10.1.1.3), to report an abstract error.

10.1.3.1 *ExplicitRN-TAPDU*

The ExplicitRN-TAPDU comprises following elements:

{
ExplicitRN-TAPDU
ExplicitRN-TAPDU ::= SEQUENCE { |

Send
[0]

Send **::=** **S**
[0]
expli-
citRN
ExplicitRNTAP-
DUId,

Send
[1] SET {
|

Send **::=**
send
[0] SEQUENCE {
|

Send **::=** **S**
[0]
reci-
pients
RecipientsElemen-
tId,

Send **::=** **S**
[0]
recipient-name
ORName }

Send **::=**
send
[1] SEQUENCE {
|

Send **::=** **S**
[0]
prior-
ity
PriorityElemen-
tId,

Send **::=** **S**
[0]
priority-ind
PriorityValue DEFAULT normal } |
PTIONAL,

Send **::=**
send
[2] SEQUENCE {
|

Send **::=** **S**
[0]
subjec-
tIPM

tId,

[0]

PTIONAL,

send

|

[0]

tor

tId,

[0]

PTIONAL,

send

|

[0]

ceipt

tId,

[0]

PTIONAL,

send

|

[0]

Types

tId,

[5]

[0]

SubjectIPMElemen-

Send **::=** **S**

subject-ipm-id
IPMIdentifier } **|**

Send **::=**

[3] SEQUENCE {

Send **::=** **S**

IPNOrigina-

IPNOriginatorElemen-

Send **::=** **S**

ipn-originating-user
ORDescriptor } **|**

Send **::=**

[4] SEQUENCE {

Send **::=** **S**

timeOfRe-

TimeOfReceiptElemen-

Send **::=** **S**

receipt-time
DateandTime } **|**

Send **::=**

[5] SEQUENCE {

Send **::=** **S**

convertedInfo-

ConvertedInfoTypesElemen-

Send **::=** **send**

SET OF

Send **::=** **S**

eIT

EITValue }	PTIONAL }
}	
}	
{	
Note — If receipt-time element defined in Receipt is omitted, TLMAU extracts one from the CES of the session in which this TAPDU was transferred. This may differ from the time of actual receipt of IPM.	
}	

Tableau [T26.330], p.

10.1.3.2 *ExplicitRNAck-TAPDU*

The ExplicitRNAck-TAPDU comprises following elements:

H.T. [T27.330]

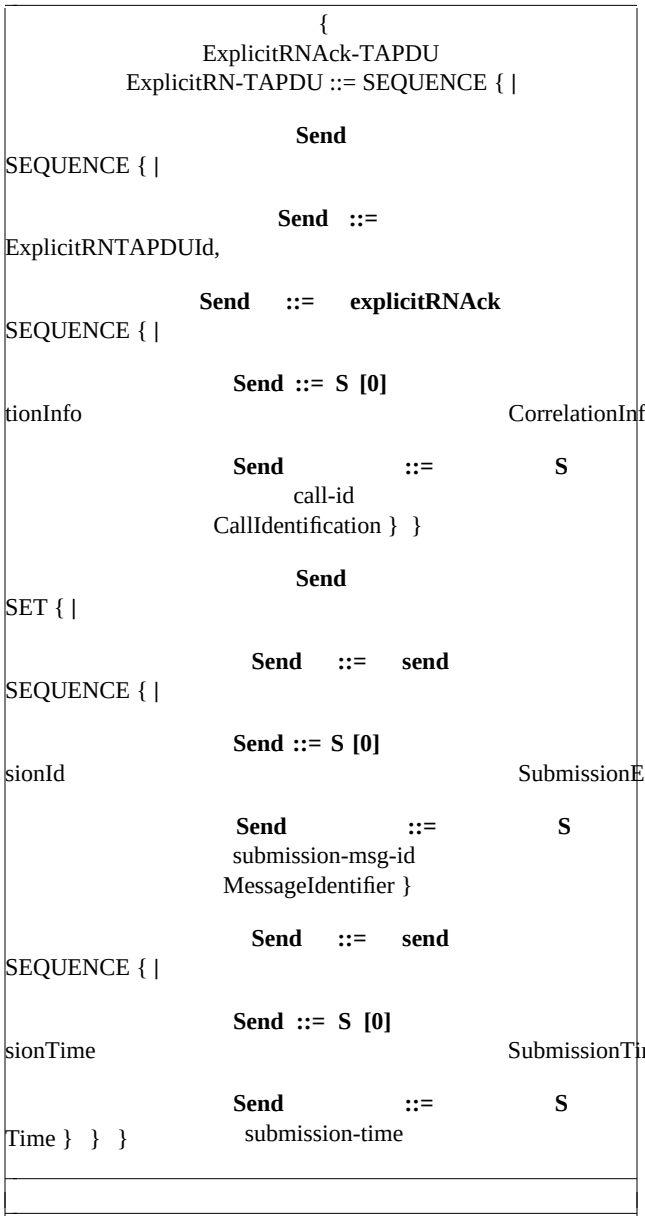

Tableau [T27.330], p.

10.1.4 *MessageCancel*

A TLM terminal sends a Cancel-TAPDU to invoke the MessageCancel abstract operation. The TLMAU returns no TAPDU to report the result of that operation, or may return an Exception-TAPDU (see § 10.1.1.3), to report an abstract error.

10.1.4.1 *Cancel-TAPDU*

The Cancel-TAPDU comprises following elements:

H.T. [T28.330]

{
Cancel-TAPDU
Cancel-TAPDU ::= SEQUENCE { |

Send ::= cancel [0] CancelTAPDUId,

Send ::= cancel [1] SEQUENCE { |

Send ::= S [0] submissionId SubmissionIdElementId,

Send ::= S [0]
submission-msg-id
MessageIdentifier } | PTIONAL,

Send ::= cancel [2] SEQUENCE { |

Send ::= S [0]
correlation-Info
Correlation-InfoElementId,

Send ::= S [0]
call-id
CallIdentification } | PTIONAL }

-- one of these must be present
}

Tableau [T28.330], p.

10.1.5 *MessageDeliver*

A TLMAU sends a Deliver-TAPDU to invoke the MessageDeliver abstract operation.

10.1.5.1 *Deliver-TAPDU*

The Deliver-TAPDU comprises following elements:

{
Deliver-TAPDU
Deliver-TAPDU ::= SEQUENCE { |

Send [0]
SEQUENCE { |

Send ::= deliver [0]
DeliverTAPDUId,

Send **::=**
deliver [1]
SEQUENCE { |

Send ::= S [0] quanti-
tyOfDocs QuantityOfDocsElementId,

Send **::=** **S** **[0]**
number-of-docs NumberOfAssociatedDo-
cuments } | PTIONAL }

-- MTS parameters

Send [1]
SET { |

Send **::=**
deliver [0]
SEQUENCE { |

Send ::= S [0] prior-
ity PriorityElementId,

Send **::=** **S** **[0]**
priority-ind PriorityValue DEFAULT
normal } | PTIONAL,

Send **::=**
deliver [1]
SEQUENCE { |

Send ::= S [0] origi-
nator OriginatorElementId,

Send **::=** **S** **[0]**
originator-name ORName } |
PTIONAL,

Send **::=**
deliver [2]
SEQUENCE { |

Send ::= S [0] thisRe-
cipient ThisRecipientElementId,

Send **::=** **S** **[0]**
this-recipient-name
ORName }

Send **::=**
deliver [3]
SEQUENCE { |

Send ::= S [0] orgInten-
dedRecipient OrgIntendedRecipientElementId,

Send ::= S [0]
org-intended-recipient-name ORName } |
PTIONAL,

Send ::=
deliver [4]
SEQUENCE { |

Send ::= S [0] otherRe-
cipients OtherRecipientsElementId,

Send ::= deliver [4]
SEQUENCE { | fR SET
OF

Send ::= S [0]
otherRecipient-name ORName } |
PTIONAL,

Send ::=
deliver [5]
SEQUENCE { |

Send ::= S [0] redirec-
tedfrom RedirectedFromElementId,

Send ::= deliver [5]
SEQUENCE { | fR SEQUENCE
OF

Send ::= S [0]
redirected-from ORName } |
PTIONAL,

Send ::=
deliver [6]
SEQUENCE { |

Send ::= S [0] submis-
sionTime SubmissionTimeElementId,

Send ::= S [0]
submission-time Dateand-
Time }

Send ::=
deliver [7]
SEQUENCE { |

Send ::= S [0]
deliveryId
DeliveryElementId,

Send ::= S [0]
delivery-msg-id
MessageIdentifier } | PTIONAL,

Send ::=
deliver [8]
SEQUENCE { |

Send ::= S [0] conver-
sionIndication ConversionIndicationElementId,

Send ::= deliver [8]
SEQUENCE { | fR SET {

Send ::= deliver [8] SEQUENCE { | ET { | fR [0] SET OF
Send ::= S [0] eIT EITValue OPTIONAL } }
Send ::= S [0] conversion-prohibited [1] ConversionProhibitedValue OPTIONAL } } | PTIONAL,
Send ::= deliver [9] SEQUENCE { |
Send ::= S [0] convertedInfoTypes ConvertedInfoTypesElementId,
Send ::= deliver [9] SEQUENCE { | fR SET OF
Send ::= S [0] eIT EIT-Value } } |
}

Tableau [1T29.330], p.12

H.T. [2T29.330]

{
Deliver-TAPDU *(continued)*
-- Deliver-TAPDU Definition (continued)

-- IPMS parameters

Send [2]
SET { |

Send ::=
deliver [0]
SEQUENCE { |

Send ::= S [0]
thisIPM ThisIP-
MElementId,

Send ::= S [0]
this-ipm-id
IPMIdentifier }

Send ::=
deliver [1]
SEQUENCE { |

Send ::= S [0]
from
FromElementId,

Send ::= S [0]
originating-user OrDescriptor } |
PTIONAL,

Send ::=
deliver [2]
SEQUENCE { |

Send ::= S [0] author-
izing AuthorizingElementId,

Send ::= deliver [2] SEQUENCE { | fR SET OF

Send ::= S [0]
authorizing-user ORDescriptor }
| PTIONAL,

Send ::=
deliver [3]
SEQUENCE { |

Send ::= S [0]
to
ToElementId,

Send ::= deliver [3] SEQUENCE { | fR SET OF SEQUENCE { |

Send ::= S [0] primary-recipient
[0] ORDescriptor,

Send ::= S [0]
primary-recipient [1]
NotificationSpecification } } | PTIONAL,

Send ::= deliver [4] SEQUENCE { |

Send ::= S [0] cc CCElementId,

Send ::= deliver [4] SEQUENCE { | fR SET OF SEQUENCE { |

Send ::= S [0] copy-recipient [0] ORDescriptor,

Send ::= S [0] copy-recipient [1] NotificationSpecification } } | PTIONAL,

Send ::= deliver [5] SEQUENCE { |

Send ::= S [0] bcc BCCElementId,

Send ::= deliver [5] SEQUENCE { | fR SET OF SEQUENCE { |

Send ::= S [0] blind-copy-recipient [0] ORDescriptor,

Send ::= S [0] blind-copy-recipient [1] NotificationSpecification } } | PTIONAL,

Send ::= deliver [6] SEQUENCE { |

Send ::= S [0] replied-ToIPM RepliedToIPMElementId,

Send ::= S [0] replied-to-ipm-id IPMIdentifier } | PTIONAL,

Send ::= deliver [7] SEQUENCE { |

Send ::= S [0] obsoletedIPMs ObsoletedIPMsElementId,

Send ::= deliver [7] SEQUENCE { | fR SET OF

Send ::= S [0] obsoleted-ipm-id IPMIdentifier } | PTIONAL,

Send deliver	**::=** [8] SEQUENCE { |
Send ::= S [0]	relatedIPMs RelatedIPMsElementId,
Send ::= deliver [8] SEQUENCE { | fR SET OF	
Send ::= S [0] related-ipm-id	IPMIdentifier } | PTIONAL,
Send deliver	**::=** [9] SEQUENCE { |
Send ::= S [0]	subject SubjectElementId,
Send ::= S [0] subject-content	SubjectContent } | PTIONAL,
Send ::= deliver	[10] SEQUENCE { |
Send ::= S [0]	contentIndicator ContentIndicatorElementId,
Send ::= deliver [10] SEQUENCE { | fR	**SEQUENCE { |**
Send ::= S [0] expiry-time	[0] DateandTime OPTIONAL,
expiry-time	[1] SET { |
Send ::= S [0]	importance [0] ImportanceValue DEFAULT normal,
Send ::= S [0]	sensitivity [1] SensitivityValue OPTIONAL,
Send ::= S [0] auto-forwarded	[2] AutoForwarded-Value DEFAULT
auto-forwarded [2]	notAutoForward } } } | PTIONAL,
	}

Tableau [2T29.330], p.13

H.T. [3T29.330]

{
Deliver-TAPDU *(end)*
-- Deliver-TAPDU Definition (continued)

Send ::= deliver [11]
SEQUENCE { |

Send ::= S [0]
reply
ReplyElementId,

Send ::= deliver [11]
SEQUENCE { | fR SEQUENCE { |

Send ::= S [0]
reply-time [0]
DateandTime,

reply-time [1] SET
OF

Send ::= S [0]
reply-recipient ORDescriptor }
} | PTIONAL,

Send ::= deliver [12]
SEQUENCE { |

Send ::= S [0]
language Langu-
ageElementId,

Send ::= S [0]
language-ind LanguageInd } |
PTIONAL,

MsgIncomplete [13] MsgIncom-
pleteElementId OPTIONAL }

-- Body

Send [3]
SEQUENCE { |

Send ::=
deliver [0]
SEQUENCE { |

Send ::= S [0] body-
Type BodyTypeElementId,

Send ::= S [0]
body-part BodyPartValue }
| PTIONAL,

Send ::=
deliver [1]
SEQUENCE { |

Send ::= S [0] forwar-
dedInfo ForwardedInfoElementId,

Send ::= deliver [1]
SEQUENCE { | fR SEQUENCE {
|

Send ::= S [0] forwarded-time [0] DateandTime,

Send ::= S [0] forwarded-time [1] DeliveryEnvelope } } | PTIONAL,

-- Delivery Envelope contains same set of MTS parameters of Deliver-TAPDU }
} *-- Definition of Notification Specification*

-- Notification Specification ::= SET { |

-- Notification Specification ::= SET { | fR rn-request [0] RNRequest-Value OPTIONAL,

-- Notification Specification ::= SET { | fR nrn-request [1] NRNRequest-Value OPTIONAL,

-- Notification Specification ::= SET { | fR return-request [2] ReturnRequest-Value OPTIONAL,

-- Notification Specification ::= SET { | fR reply-request [3] ReplyRequest-Value DEFAULT

-- Notification Specification ::= SET { | fR reply-request [3] noReply }
}

Tableau [3T29.330], p.14

10.1.6 *ReceiptStatusNotice*

A TLMAU terminal sends a ReceiptStatusNotice-TAPDU to invoke the ReceiptStatusNotice abstract operation.

10.1.6.1 *ReceiptStatusNotice-TAPDU*

The ReceiptStatuNotice-TAPDU comprises following elements:

H.T. [1T30.330]

{
ReceiptStatusNotice-TAPDU
ReceiptStatusNotice-TAPDU ::= SEQUENCE { |

Send [0]
SEQUENCE { |

Send ::= receiptStatusNotice [0]
ReceiptStatusNoticeTAPDUId,

Send ::= **receiptSta-**
tusNotice [1] SEQUENCE { |

Send ::= S [0] quanti-
tyOfDocs QuantityOfDocsElementId,

Send ::= S [0]
number-of-docs NumberOfAssociatedDo-
cuments } | PTIONAL }

-- MTS parameters

Send [1]
SET { |

Send ::=
send [0]
SEQUENCE { |

Send ::= S [0] prior-
ity PriorityElementId,

Send ::= S [0]
priority-ind Priori-
tyValue }

Send ::=
send [1]
SEQUENCE { |

Send ::= S [0]
deliveryId
DeliveryIdElementId,

Send ::= S [0]
delivery-id
MessageIdentifier } | PTIONAL,

Send ::=
send [2]
SEQUENCE { |

Send ::= S [0] origi-
nator OriginatorElementId,

Send ::= S [0]
originator-name ORName } |
PTIONAL,

Send ::=
send [3]
SEQUENCE { |

Send ::= S [0] thisRe-
cipient ThisRecipientElementId,

Send ::= S [0]
this-recipient-name
ORName }

Send ::=
send [4]
SEQUENCE { |

Send ::= S [0] submis-
sionTime SubmissionTimeElementId,

Send ::= S [0] submis-
sionTime DateandTime }

Send ::=
send [5]
SEQUENCE { |

Send ::= S [0] timeOf-
Delivery TimeOfDeliveryElementId,

Send ::= S [0]
delivery-time Dateand-
Time }

Send ::=
send [6]
SEQUENCE { |

Send ::= S [0] conver-
sionIndication ConversionIndicationElementId,

Send ::= send [6]
SEQUENCE { fR SET {
|

Send ::= S [0] **conver-**
sionprohibited [0] SET OF

Send ::= S [0]
eIT EITValue } |
PTIONAL } }

Send ::= S [0]
conversion-prohibited [1] ConversionProhibitedValue
OPTIONAL } } | PTIONAL,

Send ::=
send [7]
SEQUENCE { |

Send ::= S [0] conver-
tedInfoTypes ConvertedInfoTypesElementId,

Send ::= send [6]
SEQUENCE { fR SET
OF

Send ::= S [0]
eIT EIT-
Value } } |

-- IPMS parameters

Send	[2] SET { |
Send	**::=** **send** [0] SEQUENCE { |
Send	**::= S [0]** notificationType NotificationTypeElementId,
Send	**::= S [0]** report-type Report-TypeValue }
Send	**::= send** [1] SEQUENCE { |
Send ::= S [0]	subjec-tIPM SubjectIPMElementId,
Send	**::= S [0]** subject-ipm-id IPMIdentifier }
Send	**::= send** [2] SEQUENCE { |
Send ::= S [0]	IPNOri-ginator IPNOriginatorElementId,
Send	**::= S [0]** ipn-originating-user ORDescriptor } | PTIONAL,
Send	**::= send** [3] SEQUENCE { |
Send ::= S [0]	prefer-redRecipient PreferredRecipientElementId,
Send	**::= S [0]** preferred-recipient ORDescriptor } | PTIONAL,
	}

Tableau [1T30.330], p.15

{
ReceiptStatusNotice-TAPDU *(end)*
-- ReceiptStatusNotice-TAPDU Definition (continued)

Send ::=
send [4]
SET { |

Send ::= send [4]
SET { | fR [0] SEQUENCE { |

Send ::= S [0]
timeOfReceipt TimeOfRe-
ceiptElementId,

Send ::= S [0]
receipt-time Dateand-
Time }

Send ::= send [4]
SET { | fR [1] SEQUENCE { |

Send ::= S [0]
typeOfReceipt TypeOfRe-
ceiptElementId,

Send ::= S [0]
type-of-receipt TypeOfReceiptValue
DEFAULT manual } | PTIONAL,

Send ::= send [4]
SET { | fR [2] SEQUENCE { |

Send ::= S [0] supplRe-
ceiptInfo SupplReceiptInfoElementId,

Send ::= S [0]
suppl-receipt-info SuplementaryInformation } |
PTIONAL } | PTIONAL,

Send
::=
send [5]
SET { |

Send ::= send [5]
SET { | fR [0] SEQUENCE { |

Send ::= S [0] nonRe-
ceiptInfo NonReceiptInfoElementId,

Send ::= send [5]
SET { | 1] SET { |

Send ::= S [0]
non-receipt-reason [0] NonReceip-
tReasonValue,

Send ::= S [0]
discard-reason [1] DiscardReasonValue
OPTIONAL } } | PTIONAL,

Send ::= send [5]
SET { | fR [1] SEQUENCE { |

Send ::= S [0] com-
ments CommentElementId,

Send ::= S [0]	com-
ments	Comment }
messageReturnedInd [2]	MessageReturnedIn-
dElementId OPTIONAL } } }	
	}

Tableau [2T30.330], p.16

10.1.7 *DeliveryStatusNotice*

A TLMAU terminal sends a DeliveryStatusNotice-TAPDU to invoke the DeliveryStatusNotice abstract operation.

10.1.7.1 *DeliveryStatusNotice-TAPDU*

The DeliveryStatuNotice-TAPDU comprises following elements:

{
DeliveryStatusNotice-TAPDU
DeliveryStatusNotice-TAPDU ::= SEQUENCE { |

Send [0]
SEQUENCE { |

Send ::= deliveryStatusNotice | 0]
DeliveryStatusNoticeTAPDUId,

Send **::=**
receiptStatusNotice [1]
SEQUENCE { |

Send ::= S [0] quanti-
tyOfDocs QuantityOfDocsElementId,

Send **::=** **S** **[0]**
number-of-docs NumberOfAssociated-
Documents } | PTIONAL }

Send **::=**
receiptStatusNotice [2]
SEQUENCE { |

Send ::= S [0] correla-
tionInfo CorrelationInfoElementId,

Send **::=** **S** **[0]**
call-id
CallIdentification } }

Send [1]
SET { |

Send **::=**
send [0]
SEQUENCE { |

Send ::= S [0] sub-
missionId SubmissionIdElemen-
tId,

Send **::=** **S** **[0]**
submission-msg-id
MessageIdentifier } | PTIONAL,

Send **::=**
send [1]
SEQUENCE { |

Send ::= S [0] pro-
beId ProbeIdElementId,

Send **::=** **S** **[0]**
submission-msg-id
MessageIdentifier } | PTIONAL,

Send **::=**
send [2] SET
OF { |

Send ::= **send [2]**
SET OF { | fR [0] SEQUENCE {
|

Send ::= S [0] reportedRecipient ReportedRecipientElementId,

Send ::= S [0] reported-recipient-name ORName }

Send ::= send [2] SET OF { | fR [1] SEQUENCE { |

Send ::= S [0] notificationType NotificationTypeElementId,

Send ::= S [0] report-type ReportTypeValue }

Send ::= send [2] SET OF { | fR [2] SEQUENCE { |

Send ::= S [0] intendedRecipient IntendedRecipientElementId,

Send ::= S [0] intended-recipient-name ORName }

Send ::= send [2] SET OF { | fR [3] SEQUENCE { |

Send ::= S [0] convertedInfoTypes ConvertedInfoTypesElementId,

Send ::= send [2] SET OF { | [0] SET OF

Send ::= S [0] eIT EITValue }

Send ::= send [2] SET OF { | fR [4] SET { | *-- In case of Delivery Notification,*

-- this set of element shall be present

Send ::= receiptStatusNotice [4] [0] SEQUENCE { |

Send ::= S [0] timeOfDelivery TimeOfDeliveryElementId,

Send ::= S [0] delivery-time DateandTime }

Send ::= receiptStatusNotice [4] [1] SEQUENCE { |

Send ::= S [0] typeOfUA TypeO-fUAElementId, **Send ::= S [0]** type-of-ua TypeOfUA DEFAULT public } | PTIONAL, **Send ::= receiptSta-tusNotice [4]** [2] SEQUENCE { | **Send ::= S [0]** sup-plInfo SupplInfoElementId, **Send ::= S [0]** suppl-info SuplementaryInfor-mation } | PTIONAL, **Send ::= send [2] SET OF { | fR [5] SET { |** *-- In case of Non Delivery Notification,* *-- this set of element shall be present* **Send ::= receiptSta-tusNotice [4]** [0] SEQUENCE { | **Send ::= S [0]** non-DeliveryReason NonDeliveryReasonElemen-tId, **Send ::= receiptSta-tusNotice [4] [0]** SET { | **Send ::= S [0]** reason-code [0] Reason-CodeValue, **Send ::= S [0]** diagostic-code [1] DiagnosticCode-Value OPTIONAL } } } OPTIONAL } contentReturned [3] ContenReturnedElementId OPTIONAL } } }

Tableau [T31.330], p.

10.1.8 *Register*

A TLM terminal sends a Register-TAPDU to invoke the register abstract operation. The TLMAU returns a RegisterAck-TAPDU, if necessary, to report the result of that operation, or may return an Exception-TAPDU (see § 10.1.1.3) to report an abstract error.

10.1.8.1 *Register-TAPDU*

The Register-TAPDU comprises following elements:

H.T. [T32.330]

{
Register-TAPDU
Register-TAPDU ::= SEQUENCE { |

Send [0]

Send ::= S [0] regis-
ter TAPDUIdValue,

Send [1]
SET { |

Send
[1] [0]
SET { |

Send **[1]**
[0] [0] SEQUENCE { |

Send ::= S [0] expired-
Discard ExpiredDiscardElementId,

Send ::= S [0]
discard-ipm DiscardValue DEFAULT
discard } | PTIONAL,

Send **[1]**
[0] [1] SEQUENCE { |

Send ::= S [0]
obsoleteDiscard ObsoleteDis-
cardElementId,

Send ::= S [0]
discard-ipm DiscardValue DEFAULT
discard } | PTIONAL }

Send
[1] [1]
SET { |

Send **[1]**
[1] [0] SEQUENCE { |

Send ::= S [0]
autoFWDIPMs
AutoFWDIPMsElementId,

Send ::= S [0]
auto-fwd-ipms AutoFWDIPMsValue
DEFAULT not-auto-forward } **Send ::= S**
[0] **auto-fwd-imps**
OPTIONAL,

Send **[1]**
[1] [1] SEQUENCE { |

Send ::= S [0]
autoFWDRecipients AutoFWDReci-
pientsElementId,

Send **[1] [1]**
[1] SET OF { |

Send ::= S [0]
auto-fwd-recipient-name ORName } } |
PTIONAL,

Send [1]
[1] [2] SEQUENCE { |

Send ::= S [0] autoFWD-
Heading AutoFWDHeadingElementId,

Send ::= S [0]
auto-fwd-heading AutoFWDHead-
ing } | PTIONAL,

-- For further study

Send [1]
[1] [3] SEQUENCE { |

Send ::= S [0] autoFWD-
Comment AutoFWDCommentElementId,

Send ::= S [0]
auto-fwd-comment AutoFWDCom-
ment } | PTIONAL }

Send [2]
SET { |

Send [1]
[2] [0] SEQUENCE { |

Send ::= S [0]
dsMode DSMo-
deElementId,

Send ::= S [0]
ds-mode DSModeValue }
| PTIONAL,

Send [1]
[2] [1] SEQUENCE { |

Send ::= S [0] tLMAUO-
peration TLMAUOperationElementId,

Send [1] [2]
[1] SET { |

Send ::= S [0]
error-recovery-mode [0] ErrorRecoveryMo-
deValue OPTIONAL,

Send ::= S [0]
auto-acknowledgment [1] AutoAck-
nowledgment DEFAULT manual } } |
PTIONAL,

Send [1]
[2] [2] SEQUENCE { |

Send ::= S [0] supplReci-
pientInfo SupplRecipientInfoElementId,

Send ::= S [0]
suppl-recipient-info

SuplementaryInformation } | PTIONAL,	
Send	**[1]**
[2]	[3] SEQUENCE { |
Send ::= S [0] autoOutput	AutoOutput-tElementId,
Send	**[1] [2] [3]** SET { |
Send ::= S [0] frequency	[0] Frequency OPTIONAL,
Send ::= S [0] output-time	[1] DateandTime OPTIONAL } } | PTIONAL,
Send	**[1]**
[2]	[4] SEQUENCE { |
Send ::= S [0] messageDeleteMode	MessageDeleteModeElemen-tId,
Send ::= S [0] message-delete-mode	MessageDeleteMo-deValue DEFAULT **Send ::= S [0] message-delete-mode** auto-delete } | PTIONAL } } } |
	}

Tableau [T32.330], p.

10.1.8.2 *RegisterAck-TAPDU*

The RegisterAck-TAPDU comprises following elements:

H.T. [T33.330]

{				
RegisterAck-TAPDU				
RegisterAck-TAPDU ::=				
Send	**::=**	**S**	**[0]**	
registerAck				RegisterAck-TAPDUId
				}

Tableau [T33.330], p.

10.1.9 *DSList*

A TLM terminal sends a DSQuery-TAPDU to invoke the DSList abstract operation. The TLMAU returns a DSReport-TAPDU to report the result of that operation, or may return an Exception-TAPDU (see § 10.1.1.3) to report an abstract error.

10.1.9.1 *DSQuery-TAPDU*

The DSQuery-TAPDU comprises following elements:

H.T. [T34.330]

{				
DSQuery-TAPDU				
DSQuery-TAPDU ::=				
Send	**::=**	**S**	**[0]**	
dsQuery				
DSQueryTAPDUId				
				}

Tableau [T34.330], p.

10.1.9.2 *DSReport-TAPDU*

The DSReport-TAPDU comprises following elements:

H.T. [T35.330]

{
DSReport-TAPDU
DSReport-TAPDU ::= SEQUENCE { |

Send [0]

Send ::= S [0]
dsReport DSReport-TAPDUId,

Send [1] SET OF { |

Send **[1] SET OF { | fR** **[0] SEQUENCE { |**

Send ::= S [0]
retrievalId RetrievalIdElementId,

Send ::= S [0]
retrieval-id RetrievalIdentifier }

Send **[1] SET OF { | fR** **[1] SEQUENCE { |**

Send ::= S [0] messageType MessageTypeElementId,

Send ::= S [0]
message-type MessageTypeValue }

Send **[1] SET OF { | fR** **[2] SEQUENCE { |**

Send ::= S [0] originator OriginatorElementId,

Send ::= S [0]
originator-name ORName } | PTIONAL,

Send **[1] SET OF { | fR** **[3] SEQUENCE { |**

Send ::= S [0] priority PriorityElementId,

Send ::= S [0]
priority-ind PriorityValue DEFAULT normal } | PTIONAL,

Send **[1] SET OF { | fR** **[4] SEQUENCE { |**

Send ::= S [0] messageLength MessageLengthElementId,

Send ::= S [0]
message-length MessageLength } | PTIONAL } }

}

Tableau [T35.330], p.

10.1.10 *DSDelete*

A TLM terminal sends a MessageDelete-TAPDU to invoke the DSDelete abstract operation. The TLMAU returns no TAPDU to report the result of that operation, or may return an Exception-TAPDU (see § 10.1.1.3) to report an abstract error.

10.1.10.1 *MessageDelete-TAPDU*

The MessageDelete-TAPDU comprises following elements:

H.T. [T36.330]

{
MessageDelete-TAPDU
MessageDelete-TAPDU ::= SEQUENCE { |

Send ::= S [0] messageDelete [0] MessageDeleteTAPDUId,

Send ::= S [0] **messageDelete** | 1] SEQUENCE { |

Send ::= S [0] messageSelector MessageSelectorElementId,

Send
[1] SET OF { |

Send ::= S [0]
retrieval-id
RetrievalIdentifier } } }
}

Tableau [T36.330], p.

10.1.11 *DSFetch*

A TLM terminal sends an OutputRequest-TAPDU to invoke the DSFetch abstract operation. The TLMAU returns an OutputMessage-TAPDU to report the result of that operation, or may return an Exception-TAPDU (see § 10.1.1.3) to report an abstract error.

The OutputMessage-TAPDU is sent by TLMAU to be output the message from DS. This TAPDU is triggered by one of the following events:

1) some rule (not defined in this Recommendation) which causes TLMAU to establish a connection to the TLM terminal and to send a message at a specific time, for example, the TLM terminal has registered its times of availability with TLMAU;

2) the TLM terminal establishes a connection to TLMAU and initiates a CSCC which is taken as an implicit request for output by TLMAU;

3) receipt of an OutputRequest-TAPDU.

10.1.11.1 *OutputRequest-TAPDU*

The OutputRequest-TAPDU comprises following elements:

H.T. [T37.330]

{
OutputRequest-TAPDU
OutputRequest-TAPDU ::= SEQUENCE { |

Send [0]

Send ::= S [0] outputRequest OutputRequestTAPDUId,

Send [1] SET OF SEQUENCE { |

Send
[1] [0] SEQUENCE { |

Send ::= S [0]
retrievalId
RetrievalIdElementId,

Send ::= S [0]
retrieval-id
RetrievalIdentifier }

Send
[1] [1] SEQUENCE { |

Send ::= S [0] deleteAfterOutput DeleteAfterOutputElementId,

Send ::= S [0]
delete-after-output DeleteAfterOutputValue } | PTIONAL } }
}

Tableau [T37.330], p.

10.1.11.2 *OutputMessage-TAPDU*

The OutputMessage-TAPDU comprises following elements:

H.T. [T38.330]

{
OutputMessage-TAPDU
OutputMessage-TAPDU ::= SEQUENCE { |

Send
[0] SEQUENCE { |

Send ::= S [0] outputMessage [0] OutputMessageTAPDUId,

Send ::= S [0]
outputMessage
| 1] SEQUENCE { |

Send ::= S [0]
quantityOfDocs
QuantityOfDocsElementId,

Send ::= S [0]
number-of-docs NumberOfAssociatedDocuments } | PTIONAL }

Send [1]
SET OF SEQUENCE { |

Send
[1]
[0] SEQUENCE { |

Send ::= S [0]
retrievalId
RetrievalIdElementId,

Send ::= S [0]
retrieval-id
RetrievalIdentifier }

Send
[1]
[1] SEQUENCE { |

Send ::= S [0]
messageType
MessageTypeElementId,

Send ::= S [0]
message-type
MessageTypeValue }

Send
[1]
[2] SEQUENCE { |

Send ::= S [0]
timeOfDelivery
TimeOfDeliveryElementId,

Send ::= S [0]
delivery-time
DateandTime }

-- *The remaining Components of this TAPDU are identical to the components in the Deliver, DeliveryStatusNotice and ReceiptStatusNotice-TAPDU. The actual components to be used*

depend upon the MessageType parameter value specified in the MessageType component }	

{
Note — The RetrievalIdentifier is an identifier which identifies a message in DS.
}	

Tableau [T38.330], p.

10.1.12 *MessageState*

A TLM terminal sends a StatusQuery-TAPDU to invoke the MessageState abstract operation. The TLMAU returns a StatusReport-TAPDU to report the result of that operation, or returns an Exception-TAPDU to report an abstract error.

10.1.12.1 *StarusQuery-TAPDU*

The StatusQuery-TAPDU comprises following elements:

H.T. [T39.330]

{ StatusQuery-TAPDU StatusQuery-TAPDU ::= SEQUENCE { | **Send** [0] **Send ::= S [0]** statusQuery StatusQueryTAPDUId, **Send** [1] SET { | **Send [1]** [0] SEQUENCE { | **Send ::= S [0]** submissionId SubmissionIdElementId, **Send ::= S [0]** submission-msg-id MessageIdentifier } | PTIONAL, -- *See Note* **Send [1]** [1] SEQUENCE { | **Send ::= S [0]** correlationInfo CorrelationInfoElementId, **Send ::= S [0]** call-id CallIdentification } | PTIONAL } } -- *See Note* }
{ *Note* — If none of these are present all outstanding (in operation), operations will be reported. }

Tableau [T39.330], p.

10.1.12.2 *StatusReport-TAPDU*

The StatusReport-TAPDU comprises following elements:

H.T. [T40.330]

{
StatusReport-TAPDU
StatusReport-TAPDU ::= SEQUENCE { |

Send [0]
SEQUENCE { |

Send ::= S [0] statusReport [0] StatusReportTAPDUId

Send ::= S [0] **statusReport** [1] SEQUENCE { |

Send ::= S [0] correlationInfo CorrelationInfoElementId,

Send ::= S [0]
call-id
CallIdentification } }

Send [1]
SET { |

Send
[1] [0]
SEQUENCE { |

Send ::= S [0]
timeOfReport TimeOfReportElementId,

Send ::= S [0]
report-time DateandTime }

Send
[1] [1]
SEQUENCE { |

Send ::= S [0] reportedMessageId ReportedMessageIdElementId,

Send ::= S [0]
reported-message-id MessageTypeValue }

Send
[1] [2] SET OF
SEQUENCE { |

Send
[1]
[2] [0]
SEQUENCE { |

Send ::= S [0] actualRecipient ActualRecipientElementId,

Send ::= S [0]
actual-recipient-name
ORName }

Send
[1]
[2] [1]
SEQUENCE { |

Send ::= S [0]	mes- sageStatus [1] MessageStatusElementId,
Send ::= S [0] status **[1]**	Sta- tusValue }
Send ::= S [0]	[2] SET { | *-- In case of DN, this set*
-- of element shall be present	
Send ::= S [0]	**[2]** **SET { | fR [0] SEQUENCE {** **|**
Send ::= S [0]	timeOf- Delivery TimeOfDeliveryElementId,
Send ::= S [0] delivery-time	Dateand- Time }
Send ::= S [0]	**[2]** **SET { | fR [1] SEQUENCE {** **|**
Send ::= S [0]	typeO- fUA TypeOfUAElementId,
Send ::= S [0] type-of-ua	TypeOfUA DEFAULT public } | PTIONAL } | PTIONAL }
Send ::= S [0]	[3] SEQUENCE { | *-- In case of DN, this set*
-- of element shall be present	
Send ::= S [0]	non- DeliveryReason NonDeliveryReasonElemen- tId,
Send ::= S [0] **[3]** SET { |	
Send ::= S [0] reason-code	[0] Reason- CodeValue,
Send ::= S [0] diagnostic-code [1]	DiagnosticCodeValue } | PTIONAL } } | PTIONAL } }
	}

Tableau [T40.330], p.

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:

H.T. [T41.330]
Source of MTAS MessageSubmission arguments

MessageSubmission argument
originator-name Authenticate User (remarque 1) }

{
original-encoded-information-types
}
set by TLMAU to EITs of submitted IPM's body
}

content-type
content-identifier
content-correlator
recipient-name 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. }

Tableau [T41.330], p.

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:

H.T. [T42.330]

Source of SendAck-TAPDU components

SendAck-TAPDU component Element name	Value name	Source
correlationInfo CallIdentification that identifies previous Send-TAPDU being reported on }	call-id	{
submissionId MTS message-submission-identifier }	submission-msg-id	{
submissionTime MTS message-submission-time }	submission-time	{

Tableau [T42.330], p.

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:

H.T. [T43.330]

Source of ProbeAck-TAPDU components

ProbeAck-TAPDU component Element name	Value name	Source
correlationInfo CallIdentification that identifies previous Probe-TAPDU being reported on }	call-id	{
probeId MTS probe-submission-identifier }	probe-msg-id	{
submissionTime MTS probe-submission-time }	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:

H.T. [T44.330]
Source of MTAS MessageSubmission components

MessageSubmission argument
originator-name

{
original-encoded-information-types
}
set by TLMAU to ‘‘unspecified’’
}

content-type
content-identifier
priority

per-message-indicators
disclose-recipients
set to ‘‘disclosure-of-recipient-prohibited’’
}
conversion-prohibited
set to ‘‘conversion-prohibited’’
}
{
alternate-recipient-allowed
}
set to ‘‘alternate-recipient-prohibited’’
}
{
content-return-request
}
set to ‘‘content-return-not-requested’’
}

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’’. }

Tableau [T44.330], p.

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:

H.T. [T45.330]
Source of ExplicitRNAck-TAPDU components

{ ExplicitRNAck-TAPDU component } Element name	Source Value name	
correlationInfo CallIdentification that identifies previous ExplicitRN-TAPDU being reported on }	call-id	{
submissionId MTS message-submission-identifier }	submission-msg-id	{
submissionTime MTS message-submission-time }	submission-time	{

Tableau [T45.330], p.

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:

H.T. [T46.330]
Source of CancelDeferredDelivery arguments

{	{ Operation Element name	Value name	
{ message-submission-identifier }	submissionId	submission-msg-id	

Tableau [T46.330], p.

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:

H.T. [T47.330]
Source of Deliver-TAPDU component

Deliver-TAPDU component Corresponding MessageDelivery argument } Element name
quantityOfDocs when control Informa tion is conveyed by a normal document, set number of associated documents in Deliver-TAPDU }
priority
originator
thisRecipient
intendedRecipient originally-intended-recipient-name }
otherRecipients
submissionTime
timeOfDelivery
deliveryId message-delivery-identifier }
conversionIndication original-encoded-information-types }
conversionIndication
convertedInfoTypes 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. }

Tableau [T47.330], p.

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:

H.T. [T48.330]
Source of ReceiptStatusNotice-TAPDU components

{ ReceiptStatusNotice-TAPDU component } Corresponding MessageDelivery and receive RN/NRN argument } Element name
quantityOfDocs when control Informa tion is conveyed by a normal document, set number of associated documents in ReceiptStatusNotice-TAPDU }
priority
deliveryId message-delivery-identifier }
originator if this element is omitted, this argument should be constructed from TID obtains from CSS }
thisRecipient this-recipient-identifier }
submissionTime
timeOfDelivery
conversionIndication original-encoded-information-types }
conversionIndication
convertedInfoTypes converted-encoded-information-types }
notificationType
subjectIPM
iPNOriginator
preferredRecipient
timeOfReceipt
typeOfReceipt
supplReceiptInfo
nonReceiptInfo
nonReceiptInfo
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 fro MessageDeliver operation arguments and IPMS APDU values, as indicated above. }

Tableau [T48.330], p.

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:

H.T. [T49.330]
Source of DeliveryStatusNotice-TAPDU components

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

Tableau [T49.330], p.

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.

H.T. [T50.330]

TABLE 3/T.330

The structure of TAPDUs

{

TAPDU ::= SEQUENCE { |

TAPDU ::= SEQUENCE { | fR ControlInfo,

TAPDU ::= SEQUENCE { | fR SEQUENCE OF MessageBodyInfo OPTIONAL }

ControlInfo ::= SEQUENCE { |

ControlInfo ::= SEQUENCE { | fR TAPDUId,

ControlInfo ::= SEQUENCE { | fR Elements OPTIONAL }

TAPDUId ::= SEQUENCE { |

TAPDUId ::= SEQUENCE { | fR TAPDUNumber OPTIONAL,

TAPDUId ::= SEQUENCE { | fR TAPDUName OPTIONAL }

-- One of this must be present

Elements ::= SEQUENCE { |

Elements ::= SEQUENCE { | fR ElementId,

Elements ::= SEQUENCE { | fR ElementValues }

ElementId ::= SEQUENCE { |

ElementId ::= SEQUENCE { | fR ElementNumber OPTIONAL,

ElementId ::= SEQUENCE { | fR ElementName OPTIONAL }

-- One of this must be present

ElementValues ::= SET OF Component

-- See Note

Component ::= CHOICE { |

Component ::= SEQUENCE { | fR PrimitiveComponent,

Component ::= SEQUENCE { | fR ConstructorComponent }

Primi-
tiveComponent ::= Parameter

ConstructorCom-
ponent ::= SET OF Parameter

Parameter
::= SEQUENCE { |

Parameter ::= SEQUENCE { | fR
ParameterId OPTIONAL,

Parameter ::= SEQUENCE { | fR
ParameterValue }

}
{

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

H.T. [T51.330]

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 79:□ AUTO-FWD-COMMENT: }	Primitive	{	
autoFWDHeading 78:□ AUTO-FWD-HEADING: }	for further study	{	
autoFWDIPMs	Primitive	76:□ AUTO-FWD-IPMS:	
autoFWDRecipients 77:□ AUTO-FWD-RECIPIENTS: }	Constructor	{	
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 18 :□ CONTENT-INDICATOR: }	Constructor	{	
contentInfo	Primitive	17 :□ CONTENT-INFO:	
contentReturned 72 :□ CONTENT-RETURNED-INDICATION: }	—	{	
conversion	Primitive	16 :□ CONVERSION:	
conversionIndication 42 :□ CONVERSION-INDICATION: }	Constructor	{	
convertedInfoTypes 44 :□ CONVERTED-INFORMATION-TYPES: }	Primitive	{	
correlationInfo 1 :□ CORRELATION-INFORMATION: }	Primitive	{	
deleteAfterOutput 80 :□ DELETE-AFTER-OUTPUT: }	Primitive	{	
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:	

Tableau 3/T.330 [T50.330], p.

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.

Blanc

H.T. [1T51.330]

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 79:□ AUTO-FWD-COMMENT: }	Primitive	{	
autoFWDHeading 78:□ AUTO-FWD-HEADING: }	for further study	{	
autoFWDIPMs	Primitive	76:□ AUTO-FWD-IPMS:	
autoFWDRecipients 77:□ AUTO-FWD-RECIPIENTS: }	Constructor	{	
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 18 :□ CONTENT-INDICATOR: }	Constructor	{	
contentInfo	Primitive	17 :□ CONTENT-INFO:	
contentReturned 72 :□ CONTENT-RETURNED-INDICATION: }	—	{	
conversion	Primitive	16 :□ CONVERSION:	
conversionIndication 42 :□ CONVERSION-INDICATION: }	Constructor	{	
convertedInfoTypes 44 :□ CONVERTED-INFORMATION-TYPES: }	Primitive	{	
correlationInfo 1 :□ CORRELATION-INFORMATION: }	Primitive	{	
deleteAfterOutput 80 :□ DELETE-AFTER-OUTPUT: }	Primitive	{	
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:	

Tableau 4/T.330 [1T51.330], p.37

H.T. [2T51.330]
TABLE 4/T.330 *(cont.)*

TAPDUId and ElementId name	Type	T.61 Character coding format	Remarks
orgIntendedRecipient 40 :□ INTENDED-RECIPIENT: }	Primitive	{	
iPNOriginator	Primitive	69 :□ IPN-ORIGINATOR:	
language	Primitive	53:□ LANGUAGE:	
latestDelivery	Primitive	34:□ LATEST-DELIVERY:	
messageDelete	—	3.18 :□ MESSAGE-DELETE:	
messageDeleteMode 81 :□ MESSAGE-DELETE-MODE: }	Primitive	{	
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 46 :□ NON-DELIVERY-REASON: }	Primitive	{	
nonReceiptInfo 49 :□ NON-RECEIPT-INFO: }	Primitive	{	
 43 :□ NOTIFICATION-TYPE: }		{	
obsoletedDiscard 74 :□ OBSOLETED-DISCARD: }	Primitive	{	
obsoletedIPMs	Constructor	29 :□ OBSOLETED:	
otherRecipients	Constructor	41 :□ OTHER-RECIPIENTS:	
outputMessage	—	3.10 :□ OUTPUT-MESSAGE:	
outputRequest	—	3.9 :□ OUTPUT-REQUEST:	
perMessageIndicators	Constructor	19 :□ FLAGS:	
preferredRecipient 70 :□ PREFERRED-RECIPIENT: }	Primitive	{	
priority	Primitive	13 :□ PRIORITY:	
probe	—	3.2 :□ PROBE:	
probeAck	—	3.15 :□ PROBE-ACK:	

probeId	Primitive	66 :□ PROBE-ID:	
quantityOfDocs 62 :□ QUANTITY-OF-DOCS: }	Primitive	{	
recipients	Constructeur	15 :□ RECIPIENTS:	
receiptStatusNotice 3.5 :□ RECEIPT-STATUS-NOTICE: }	—	{	
redirectedFrom	Constructeur	54 :□ REDIRECTED-FROM:	
register	—	3.11 :□ REGISTER:	
registerAck	—	3.17 :□ REGISTER-ACK:	

Tableau 4/T.330 [2T51.330], p.38

H.T. [3T51.330]
TABLE 4/T.330 *(end)*

TAPDUId and ElementId name	Type	T.61 Character coding format	Remarks
relatedIPMs	Constructor	28 :□ RELATED-IPMS:	
repliedToIPM 30 :□ REPLIED-TO-IPM: }	Primitive	{	
reply	Constructor	25 :□ REPLY:	
reportedMessageId 75 :□ REPORTED-MESSAGE-ID: }	Primitive	{	
reportedRecipient 3 :□ REPORTED-RECIPIENT: }	Primitive	{	
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 68 :□ SUPPLEMENTARY-INFORMATION: }	Primitive	{	
supplReceiptInfo 68 :□ SUPPLEMENTARY-INFORMATION: }	Primitive	{	
thisIPM	Primitive	27 :□ THIS-IPM:	
thisRecipient	Primitive	39 :□ THIS-RECIPIENT:	
timeOfDelivery 4 :□ TIME-OF-DELIVERY: }	Primitive	{	
timeOfReceipt 47 :□ TIME-OF-RECEIPT: }	Primitive	{	
timeOfReport 84 :□ TIME-OF-REPORT: }	Primitive	{	
tLMAUOperation	Constructor	59 :□ TLMAU-OPERATION:	
to	Constructor	22 :□ TO:	
typeOfReceipt	Primitive	{	

48 :□ TYPE-OF-RECEIPT: }			
typeOfUA 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) abc de: underlined characters, i.e. ‘‘abc’’ are mandatory in case of international access (see § 11.2). 4) □: space character. }	Primitive	{	

Tableau 4/T.330 [3T51.330], p.39

H.T. [1T52.330]

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	Allow ed	

authorizing	Constructor	—	21 :□AUTHORIZING:	
authorizing-user	Constructor	General	*OR Descriptor*	
auto-acknowledgment Auto-Rec eipt, Manual-Rec eipt (d´efaut) }	Primitive	Predefined	{	
auto-forwarded Auto-for warded, Not-Auto-for warded (default) }	Primitive	Predefined	{	
auto-fwd-Comment	Primitive	General	*any Text*	
auto-fwd-ipms Auto-for warded, Not-Auto-for warded (default) }	Primitive	Predefined	{	

{
auto-fwd-recipient-name
}
	Constructor	General	*OR Name*	

blind-copy-recipient	Constructor	General	*OR Descriptor*	
body-part IA5Text , TLX , Voice , G3Fax , G4Fax -Class1, TTX , Videotex , Message , Mixed -Mode, Encrypted }	Primitive	Predefined	{	
call-id	Primitive	General	*Call Identification*	
comments	Primitive	General	*Comments*	
content-return-request Content-Return-Req uest }	Primitive	Predefined	{	
conversion-info			N O, Y es; WL OSS	
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 | ua)
}
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
}
	Primitive	Predefined	{	

Tableau 5/T.330 [1T52.330], p.40

H.T. [2T52.330]
TABLE 5/T.330 *(cont.)*

Element value name	Type of element value field	Type of value	T.61 character coding format
discard-ipm Discard (default), Not-Discard }	Primitive	Predefined	{
discard-reason IPM-Expired , IPM-Obsoleted , User-Subscription-Terminated }	Primitive	Predefined	{
disclose-recipients	Primitive	Predefined	No-Discl osure
dsMode	Primitive	—	58 :□DS-MODE:
ds-mode	Primitive	Predefined	Auto-Output , Retrieval
eIT IA5Text , TLX , Voice , G3Fax , G4Fax -Class1, TTX , Videotex , Undefined , Mixed -Mode }	Primitive	Predefined	{
error-cause 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	Primitive	Predefined	{

,*
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 Recovery-1 , Recovery-2 , Recovery-3 }	Primitive	Predefined	{
expiry-time	Primitive	General	*Date and Time*
explicit-conversion TLX , IA5 , G3 , G4 , VTX , TTX }	Primitive	Predefined	{
forwarded-time	Primitive	General	*Date and Time*
frequency	Primitive	General	*Frequency*
importance L ow, N ormal (default), H igh }	Primitive	Predefined	{
intended-recipient-name	Constructor	General	*OR Name*
ipn-originating-user	Constructor	General	*OR Descriptor*
language-ind	Primitive	Predefined	
latest-delivery-time	Primitive	General	*Date and Time*
message-delete-mode Auto-Del ete (default), Manual-Del ete }	Primitive	Predefined	{
message-length	Primitive	General	*Message Length*

messageType	Primitive	—	52 :□MESSAGE-TYPE:
non-receipt-reason IPM-DISCARD , IPM-Auto-forwarded }	Primitive	Predefined	{
nrn-request	Primitive	Predefined	NRN -Request
number-of-docs *Number Of Associated Documents* }	Primitive	General	{

Tableau 5/T.330 [2T52.330], p.41

H.T. [3T52.330]
TABLE 5/T.330 *(cont.)*

Element value name	Type of element value field	Type of value	T.61 character coding format
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 ‘‘PDM=’’OM (defaut), EMS, SPEC, COL, TLXA, TTXA, PHA, BFAX }	Primitive	Predefined	{
{ Physical-delivery-report-request } ‘‘REP=’’UND (defaut), PDS, MHS, PDMHS }	Primitive	Predefined	{
{ 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 Urg ent, Non-Urg ent, Nor mal (d´efaut) }	Primitive	Predefined	{
probe-msg-id	Primitive	General	*Message Identifier*
reason-code Transfer-Failure , Unable-To-Transfer , Conversion-Not-Performed }	Primitive	Predefined	{
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 Reply , No-Reply (default) }	Primitive	Predefined	{
reply-time	Primitive	General	*Date and Time*
reported-message-id	Primitive	General	*Message Identifier*

Tableau 5/T.330 [3T52.330], p.42

H.T. [4T52.330]
TABLE 5/T.330 *(end)*

Element value name	Type of element value field	Type of value	T.61 character coding format	F
reported-recipient-name	Constructor	General	*OR Name*	
report-time	Primitive	General	*Date and Time*	
report-type Receipt , Non-Receipt , Delivery , Non-delivery }	Primitive	Predefined	{	

{
requested-delivery-method
}
‘‘RDL=’’ANY (d´efaut), MAS, PD, TLX, TTX, G3, G4, IA5, VTX
}
	Primitive	Predefined	{	

retrieval-id *Retrieval Identifier* }	Primitive	General	{	
return-request	Primitive	Predefined	Return -Request	
rn-request	Primitive	Predefined	RN -Request	
sensitivity Pers onal, Priv ate, Comp any-Confidential }	Primitive	Predefined	{	
status In-Process , Delivered , Non-Delivered }	Primitive	Predefined	{	
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 *Supplementary Information* }	Primitive	General	{	
suppl-receipt-info *Supplementary Information* }	Primitive	General	{	
terminal-type ‘‘TTyp=’’ TLX, TTX, G3, G4, IA5, vtx }	Primitive	Predefined	{	
this-recipient-name	Constructor	General	*OR Name*	
this-ipm-id	Constructor	General	*IPM Identifier*	
type-of-receipt Manual (d´efaut), Auto matic }	Primitive	Predefined	{	
type-of-ua	Primitive	Predefined	{	

Priv ate, Pub lic (d´efaut) }			
user-report-request 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. *Conven- tions:* 1) Primitive: elementary or value contains only one component. 2) Constructor: element or value contains more than one component. 3) abc de: underlined characters, i.e. ‘‘abc’’ are mandatory. }	Primitive	Predefined	{

Tableau 5/T.330 [4T52.330], p.43

H.T. [1T53.330]

TABLE 6/T.330
General value list

General value Name	Parameter name	Value attribute Code | ua)	Format/Keyword

{
Each component is separated by the character <</>>
}
{
TLMAU TID
T.61
Defined in Rec. F.200
Call identification
TLM TID
T.61
Defined in Rec. F.200
}
{
Date and time
P
YY-MM-DD-HH:mm
}
{
Document rel. No.
N
001 | | 99
}
{
Additional session rel. No.
N
01 | | 9
}			

Comments	—	P	
Date and time	—	P	YY-MM-DD-HH:mm
Frequency	—	N	In minutes

{
<IPM Identifier>::=<OR Name>‘‘/’’<Local Message ID>‘‘>’’
IPM identifier
OR name
See OR name
}
{
Local message ID
AI5
‘‘LID=’’ <Local Message ID>
}			

Message identifier	—	P	
Message length	—	N	In octets

{
Number of associated documents
}
	—	N	

{
OR name
See OR name
OR descriptor
Free form name
T.61
‘‘Free Form Name=’’ <Free Form Name | *QFN=’’<Free Form Name>
}
{
Telephone number
P
‘‘Telephone Number=’’ <Telephone Number | *QTEL=’’<Telephone Number>			

}

Tableau 6/T.330 [1T53.330] (a l'italienne), p.44

{
TABLE 6/T.330 *(cont.)*
}

General value Name	Parameter name	Value attribute Code | ua)	Format/Keyword

{
<OR Name>::=<Standard Attribute Lists><Domain Defined Attribute List>
}
{
Standard attribute list
<Standard Attribute List>::= ‘‘<’’ <**Keyword.Att**
> (‘‘;’’ <**Keyword.Att**
>)*
‘‘>’’
}
{
Country name
N/P
‘‘**Country Name**
=’’<Country Name> **|** ‘‘**CN**
=’’ <Country Name>
Default: Country of TLMAU
}
{
Administration domain name
N/P
‘‘**Administration Domain Name**
=’’ <Administration Domain Name> **|**
‘‘**ADMD**
=’’ <Administration Domain Name>
D´efaut: ADMD de la TLMAU
}
{
Network adress
N
‘‘**X121 Address**
=’’ <Network Address> **|** ‘‘**X121**
=’’ <Network Address> **|**
‘‘**Network Address**
=’’ <Network Address>
}
{
Terminal identifier
P
‘‘**Terminal ID**
=’’ <Terminal Identifier> **|** ‘‘**TID**
=’’
<Terminal Identifier>
}
{
Private domain name
N/P
‘‘**Private Domain Name**
=’’ <Private Domain Name> **|** ‘‘**PRMD**
=’’ <Private
Domain Name>
}
{
Organisation name
P
‘‘**Organization Name**
=’’ <Organization Name> **|** ‘‘**ON**
=’’
<Organization Name>			

OR name
Numeric user identifier
N
‘‘**User Agent ID**
=’’ <Numeric User Identifier> **|** ‘‘**UAID**
=’’
<Numeric User Identifier> **|** ‘‘**NUID**
=’’ <Numeric User Identifier>
}
{
SurName
P
‘‘**SurName**
=’’ <SurName> **|** ‘‘**SN**
=’’ <SurName>
}
{
Given name
P
‘‘**Given Name**
=’’ <Given Name> **|** ‘‘**GN**
=’’ <Given Name>
}
{
Initials
P
‘‘**Initials**
=’’ <Initial> **|** ‘‘**I**
=’’ <Initial>
}
{
Generation qualifier
P
‘‘**Generation Qualifier**
=’’ <Generation Qualifier> **|** ‘‘**GQ**
=’’
<Generation Qualifier>
}
{
Organization unit name
P
‘‘**Organization Unit Name**
=’’ <Organization Unit Name> **|**
‘‘**OU**
=’’ <Organization Unit Name>
}
{
Domain defined attribute listne
<Domain Defined Attributes List>::= ‘‘<DDA=’’<Type>‘‘,’’<Value>(‘‘;’’
<Type>‘‘,’’<Value>)* ‘‘>’’ **|** ‘‘<Domain Defined Attributes=’’ <Type>
‘‘,’’<Value>(‘‘;’’<Type>‘‘,’’<Value>)* ‘‘>’’
}
Type P
Value P			

Tableau 6/T.330 [2T53.330] (a l’italienne), p.45

H.T. [3T53.330]

{
TABLE 6/T.330 *(cont.)*
}

General value Name	Value attribute Parameter name	Code | ua)	Format/Keyword
Postal address			
PDS-name	P	‘‘PDSN=’’ <value>	
{			
Physical delivery country name			
}	N/P	‘‘PDCN=’’ <value>	
Postal code	N/P	‘‘PC=’’ <value>	
Physical delivery office name	P	‘‘PDNA=’’ <value>	
{			
Physical delivery office number			
}	P	‘‘PDNU=’’ <value>	
{			
Extension OR address components			
}	P	‘‘EOA=’’ <value>	
{			
Physical delivery personal name			
}	P	‘‘PNP=’’ <value>	
OR name (continued)	{		
Physical delivery organization name			
}	P	‘‘ONP=’’ <value>	
{			
Extension physical delivery address components			
}	P	‘‘EPD=’’ <value>	
Unformatted postal address	P	‘‘UPA=’’ <value>	{
Max. 6 lines, max. 30 characters per line separated by <<->>			
}			
Street address	P	‘‘STA=’’ <value>	
Post office box address	P	‘‘POB=’’ <value>	
Poste restante address	P	‘‘PRA=’’ <value>	
Unique postal name	P	‘‘UN=’’ <value>	
Local postal attributes	P	‘‘LPA=’’ <value>	

Tableau 6/T.330 [3T53.330] (a l’italienne), p.46

H.T. [4T53.330]

{
TABLE 6/T.330 *(end)*
}

General value Name	Paramete
Retrieval identifier	—
Subject	—
Supplementary information P a) N = Numeric string in T.61 character set; P = Printable string in T.61 character set; AI5 = AI5 string in T.61 character set; T.61 = T.61 string. b) Descr.Att in OR Descriptor and Keyword Att in OR Name contains a Parameter-Id and a parameter-vaue. The Parameter-Ids appear with bold characters in this table. *Note 1* — Syntactical conventions are defined as follows and the character size does not make any sense: <. | |> Represents a syntactical item, non-terminal (. | |)* Itera-tion | Exclusive or alterna-tives ‘‘abc’’ Explicit characters abc. *Note 2* — Maximum length of parameter values can be found in X.400-Series Recommenda-tions. }	—

Tableau 6/T.330 [4T53.330] (a l’italienne), p.47