

AS/NZS 4033.7:1999

ISO/IEC 10021-7:1997

ISO/IEC 10021-7:1997/Cor.1:1998

ISO/IEC 10021-7:1997/Amd.1:1998

Australian/New Zealand Standard™

**Information technology—Text
communication—Message-oriented text
interchange systems**

Part 7: Interpersonal messaging system

[ISO/IEC title: Information technology—Message Handling Systems (MHS):
Interpersonal messaging system]



Standards Australia



STANDARDS
Documents

Document Sharing Hub

AS/NZS 4033.7:1999

This Joint Australian/New Zealand Standard was prepared by Joint Technical Committee IT/1, Information Systems—Interconnection. It was approved on behalf of the Council of Standards Australia on 14 July 1999 and on behalf of the Council of Standards New Zealand on 28 June 1999. It was published on 5 August 1999.

The following interests are represented on Committee IT/1:

Australian Association of Chief Information Officers
 Australian Association of Permanent Building Societies
 Australian Bankers Association
 Australian Bureau of Statistics
 Australian Chamber of Commerce and Industry
 Australian Communications Industry Forum
 Australian Computer Society
 Australian Information Industry Association
 Australian Telecommunications Users Group
 Australian Vice-Chancellors Committee
 CSIRO Mathematical and Information Sciences
 Department of Communications and the Arts
 Department of Industry Science and Tourism (Commonwealth)
 EDS (NZ)
 Electrical Compliance Testing Association
 Telecom New Zealand
 Telstra Corporation

Review of Standards. To keep abreast of progress in industry, Joint Australian/New Zealand Standards are subject to periodic review and are kept up to date by the issue of amendments or new editions as necessary. It is important therefore that Standards users ensure that they are in possession of the latest edition, and any amendments thereto.

Full details of all Joint Standards and related publications will be found in the Standards Australia and Standards New Zealand Catalogue of Publications; this information is supplemented each month by the magazines 'The Australian Standard' and 'Standards New Zealand', which subscribing members receive, and which give details of new publications, new editions and amendments, and of withdrawn Standards.

Suggestions for improvements to Joint Standards, addressed to the head office of either Standards Australia or Standards New Zealand, are welcomed. Notification of any inaccuracy or ambiguity found in a Joint Australian/New Zealand Standard should be made without delay in order that the matter may be investigated and appropriate action taken.

This Standard was issued in draft form for comment as DR 99177.



Documents

Document Sharing Hub

Australian/New Zealand Standard™

Information technology—Text communication—Message-oriented text interchange systems

Part 7: Interpersonal messaging system

Originated in Australia as AS 4033.7—1992.
Jointly revised and designated AS/NZS 4033.7:1999.

Published jointly by:

Standards Australia
1 The Crescent,
Homebush NSW 2140 Australia

Standards New Zealand
Level 10, Radio New Zealand House,
155 The Terrace,
Wellington 6001 New Zealand

PREFACE

This Standard was prepared by the Joint Standards Australia/Standards New Zealand Committee IT/1, Information Systems—Interconnection. This Standard is identical with and has been reproduced from ISO/IEC 10021-7:1997, ISO/IEC 10021-7:1997/Cor.1:1998 and ISO/IEC 10021-7:1997/Amd.1:1998, *Information technology—Message Handling Systems (MHS): Interpersonal messaging system*, Technical Corrigendum 1 and Amendment 1: *Security error diagnostic codes*.

The objective of this Standard is to provide network designers with a definition of interpersonal messaging, a form of messaging handling tailored for ordinary interpersonal business and private correspondence.

As this Standard is reproduced from an International Standard, the following applies:

- (a) Its number does not appear on each page of text and its identity is shown only on the cover and title page.
- (b) In the source text 'this Recommendation/International Standard' should read 'this Australian/New Zealand Standard'.
- (c) A full point should be substituted for a comma when referring to a decimal marker.

© Copyright – STANDARDS AUSTRALIA/STANDARDS NEW ZEALAND

Users of Standards are reminded that copyright subsists in all Standards Australia and Standards New Zealand publications and software. Except where the Copyright Act allows and except where provided for below no publications or software produced by Standards Australia or Standards New Zealand may be reproduced, stored in a retrieval system in any form or transmitted by any means without prior permission in writing from Standards Australia or Standards New Zealand. Permission may be conditional on an appropriate royalty payment. Australian requests for permission and information on commercial software royalties should be directed to the head office of Standards Australia. New Zealand requests should be directed to Standards New Zealand.

Up to 10 percent of the technical content pages of a Standard may be copied for use exclusively in-house by purchasers of the Standard without payment of a royalty or advice to Standards Australia or Standards New Zealand.

Inclusion of copyright material in computer software programs is also permitted without royalty payment provided such programs are used exclusively in-house by the creators of the programs.

Care should be taken to ensure that material used is from the current edition of the Standard and that it is updated whenever the Standard is amended or revised. The number and date of the Standard should therefore be clearly identified.

The use of material in print form or in computer software programs to be used commercially, with or without payment, or in commercial contracts is subject to the payment of a royalty. This policy may be varied by Standards Australia or Standards New Zealand at any time.

CONTENTS

	<i>Page</i>
SECTION 1 – INTRODUCTION	1
1 Scope	1
2 Normative references	1
2.1 Open Systems Interconnection	1
2.2 Message Handling Systems	2
2.3 Directory Systems	2
2.4 Language Code	2
2.5 Character Sets	2
2.6 Telematic services	3
2.7 File Transfer	3
2.8 Open Document Architecture	3
2.9 Digital Encoding of Sound	3
3 Definitions	3
4 Abbreviations	3
5 Conventions	3
5.1 ASN.1	4
5.2 Grade	5
5.3 Terms	5
5.4 Conventions for attribute-types used in Table 5	5
SECTION 2 – ABSTRACT INFORMATION OBJECTS	5
6 Overview	5
7 Interpersonal messages	6
7.1 Heading field component types	6
7.1.1 IPM identifier	6
7.1.2 Recipient specifier	7
7.1.3 OR-descriptor	7
7.1.4 IPMS extension	8
7.2 Heading fields	8
7.2.1 This IPM	9
7.2.2 Originator	9
7.2.3 Authorizing Users	9
7.2.4 Primary Recipients	9
7.2.5 Copy Recipients	9

	<i>Page</i>
7.2.6 Blind Copy Recipients	10
7.2.7 Replied-to IPM.....	10
7.2.8 Obsolete IPMs.....	10
7.2.9 Related IPMs.....	10
7.2.10 Subject	10
7.2.11 Expiry Time	11
7.2.12 Reply Time.....	11
7.2.13 Reply Recipients	11
7.2.14 Importance	11
7.2.15 Sensitivity	11
7.2.16 Auto-forwarded.....	11
7.2.17 Extensions.....	12
7.3 Body Parts.....	12
7.3.1 Extended Body Part	12
7.3.2 Body Part Encoding	13
7.4 Standard Body Part Types.....	14
7.4.1 IA5 Text.....	15
7.4.2 G3 Facsimile	15
7.4.3 G4 Class 1	16
7.4.4 Teletex.....	16
7.4.5 Videotex	17
7.4.6 Encrypted	17
7.4.7 Message.....	18
7.4.8 Mixed-mode.....	18
7.4.9 Bilaterally Defined	18
7.4.10 Nationally Defined.....	19
7.4.11 General Text.....	19
7.4.12 File Transfer.....	20
7.4.12.1 Related Stored File parameter	20
7.4.12.2 Contents Type parameter	21
7.4.12.3 Environment parameter.....	22
7.4.12.4 Compression parameter.....	22
7.4.12.5 File Attributes parameter	22
7.4.12.5.1 Pathname attribute	23
7.4.12.5.2 Permitted actions attribute	23
7.4.12.5.3 Storage account attribute	23
7.4.12.5.4 Date and time attributes.....	23
7.4.12.5.5 Identity attributes.....	24
7.4.12.5.6 Availability attribute.....	24
7.4.12.5.7 Object size attributes	24
7.4.12.5.8 Access control attribute	24
7.4.12.5.9 Legal qualifications attribute	25
7.4.12.5.10 Private use attribute	25
7.4.12.5.11 Attribute extensions.....	25
7.4.12.6 Extensions parameter	25
7.4.12.7 File Transfer Body Part data	25
7.4.12.8 Encoded Information type.....	25
7.4.13 Voice.....	25
7.4.14 Report.....	27
7.4.15 Notification	27
7.4.16 Forwarded content	27
8 Interpersonal notifications.....	28
8.1 Common fields.....	29
8.1.1 Subject IPM	30
8.1.2 IPN Originator	30
8.1.3 IPM Intended Recipient	30

	<i>Page</i>
8.1.4 Conversion EITs	30
8.1.5 Notification Extensions	31
8.2 Non-receipt fields	31
8.2.1 Non-receipt Reason	31
8.2.2 Discard Reason	31
8.2.3 Auto-forward Comment	32
8.2.4 Returned IPM	32
8.2.5 NRN Extensions	32
8.3 Receipt fields	32
8.3.1 Receipt Time	33
8.3.2 Acknowledgment Mode	33
8.3.3 Suppl Receipt Info	33
8.3.4 RN Extensions	33
8.4 Other notification type fields	33
SECTION 3 – ABSTRACT SERVICE DEFINITION	33
9 Overview	33
10 Primary object types	34
10.1 Interpersonal Messaging System user	34
10.2 Interpersonal Messaging System	35
11 Primary port types	35
11.1 Origination	35
11.2 Reception	35
11.3 Management	35
12 Abstract operations	35
12.1 Origination abstract operations	36
12.1.1 Originate Probe	36
12.1.2 Originate IPM	36
12.1.3 Originate RN	37
12.1.4 Originate ON	37
12.2 Reception abstract operations	38
12.2.1 Receive Report	38
12.2.2 Receive IPM	38
12.2.3 Receive RN	39
12.2.4 Receive NRN	39
12.2.5 Receive ON	39
12.3 Management abstract operations	39
12.3.1 Change Auto-discard	39
12.3.2 Change Auto-acknowledgment	40
12.3.3 Change Auto-forwarding	40
13 Abstract errors	41
13.1 Subscription Error	41
13.2 Recipient Improperly Specified	41
14 Other capabilities	41
SECTION 4 – ABSTRACT SERVICE PROVISION	42
15 Overview	42
16 Secondary object types	42
16.1 Interpersonal Messaging System user agent	42
16.2 Interpersonal Messaging System message store	43
16.3 Telematic agent	43
16.4 Telex access unit	44
16.5 Physical delivery access unit	44
16.6 Message transfer system	44
17 Secondary port types	44

	<i>Page</i>
17.1 Submission.....	44
17.2 Delivery	44
17.3 Retrieval.....	44
17.4 Administration	45
17.5 Import.....	45
17.6 Export.....	45
18 User agent operation	45
18.1 State variables	45
18.2 Performance of origination operations.....	45
18.2.1 Originate Probe	46
18.2.2 Originate IPM	46
18.2.3 Originate RN.....	47
18.2.4 Originate ON.....	47
18.3 Performance of management operations	48
18.3.1 Change Auto-discard	48
18.3.2 Change Auto-acknowledgment.....	48
18.3.3 Change Auto-forwarding	48
18.4 Invocation of reception operations.....	48
18.4.1 Receive Report.....	49
18.4.2 Receive IPM.....	49
18.4.3 Receive RN	49
18.4.4 Receive NRN	49
18.4.5 Receive ON	49
18.5 Internal procedures.....	49
18.5.1 Auto-discard.....	50
18.5.1.1 Discard of IPM.....	50
18.5.1.2 Construction of NRN	50
18.5.1.3 Submission of NRN	50
18.5.2 Auto-acknowledgment	50
18.5.2.1 Construction of RN	50
18.5.2.2 Submission of RN	51
18.5.3 Auto-forwarding	51
18.5.3.1 Prevention of loops	51
18.5.3.2 Construction of IPM	51
18.5.3.3 Submission of IPM	51
18.5.3.4 Construction of NRN	52
18.5.3.5 Submission of NRN	52
19 Message Store operation	52
19.1 Binding to the IPMS-MS	52
19.1.1 MS-Bind-argument	52
19.1.2 MS-Bind-result	53
19.2 Creation of information objects	53
19.2.1 Mapping an IPMS message to an MS entry	53
19.2.2 Mapping of forwarding messages in the IPMS-MS	54
19.2.3 Presence of General-attributes in child-entries	54
19.3 Maintenance of attributes.....	55
19.4 Notification of Non-receipt.....	56
19.5 IPMS-MS abstract-operation extensions	56
19.5.1 MS-Bind extensions.....	56
19.5.2 MS-message-submission extensions.....	57
19.5.2.1 Forwarding-request extension.....	57
19.5.2.2 IPM submission options.....	57
19.5.2.3 IPM submission errors	58
19.5.3 Delete extensions	58

	<i>Page</i>
19.6 IPMS-MS attributes	58
19.6.1 Summary attributes	61
19.6.1.1 IPM Entry Type	62
19.6.1.2 IPM Synopsis	62
19.6.1.3 Body Parts Summary	63
19.6.1.4 IPM Auto-discarded	64
19.6.2 Heading attributes	64
19.6.2.1 Heading	64
19.6.2.2 Heading analyses	64
19.6.2.3 Heading fields	65
19.6.2.4 Heading sub-fields	66
19.6.2.5 Heading extensions	67
19.6.3 Body attributes	67
19.6.3.1 Body	67
19.6.3.2 Extended Body Part Types	68
19.6.3.3 Extended Body Parts	68
19.6.3.4 Basic Body Parts	69
19.6.3.5 Basic Body Part Parameters components	70
19.6.3.6 Basic Body Part Data components	71
19.6.4 Notification attributes	71
19.6.4.1 Common fields	71
19.6.4.2 Non-receipt fields	72
19.6.4.3 Receipt fields	72
19.6.4.4 Other Notification Type Fields	73
19.6.5 Correlation attributes	73
19.6.5.1 Common attributes	73
19.6.5.1.1 AC Forwarding IPMs	73
19.6.5.1.2 AC Forwarded IPMs	74
19.6.5.1.3 AC Obsoleteing IPMs	74
19.6.5.1.4 AC Obsoleteed IPMs	74
19.6.5.1.5 AC Relating IPMs	75
19.6.5.1.6 AC Related IPMs	75
19.6.5.1.7 AC Replied-to IPM	75
19.6.5.1.8 AC Replying IPMs	75
19.6.5.1.9 AC Subject IPM	76
19.6.5.2 Submitted message correlation	76
19.6.5.2.1 AC IPM Recipients	76
19.6.5.2.2 AC Delivered Replies Summary	76
19.6.5.2.3 AC Correlated Delivered Replies	77
19.6.5.2.4 AC Delivered IPN Summary	77
19.6.5.2.5 AC Correlated Delivered IPNs	78
19.6.5.3 Delivered message correlation	78
19.6.5.3.1 AC Submitted Reply Status	79
19.6.5.3.2 AC Submitted IPN Status	79
19.6.5.3.3 AC Submitted IPNs	80
19.6.5.3.4 Recipient Category	80
19.6.5.3.5 Revised Reply Time	80
19.6.6 The IPMS-attribute-table information object class	81
19.6.7 Generation of the IPMS-specific attributes	81
19.6.8 Attributes subject to modification	84
19.7 IPMS-MS matching rules	85
19.7.1 IPM-identifier-match	85
19.7.2 IPM-location-match	85
19.7.3 OR-descriptor-match	85
19.7.4 OR-descriptor-elements-match	86
19.7.5 OR-descriptor-substring-elements-match	86

	<i>Page</i>
19.7.6 OR-descriptor-single-element-match	86
19.7.7 Recipient-specifier-match	86
19.7.8 Recipient-specifier-elements-match	86
19.7.9 Recipient-specifier-substring-elements-match	87
19.7.10 Recipient-specifier-single-element-match	87
19.8 IPMS-MS auto-actions	87
19.8.1 Auto-action performance	88
19.8.2 IPM auto-forward	88
19.8.3 IPM auto-acknowledgement	91
19.8.4 IPM auto-correlate	92
19.8.5 IPM auto-discard	92
19.9 Procedures for the IPMS-MS	93
19.9.1 Additional procedures for Message-delivery and Report-delivery	93
19.9.1.1 Additional procedures for IPM auto-correlate	93
19.9.1.2 Additional procedures for IPM auto-forward	94
19.9.1.3 Additional procedures for IPM auto-discard	96
19.9.2 Additional procedures for MS-message-submission	97
19.9.3 Additional procedures for Fetch	98
19.9.4 Additional procedures for Delete and Auto-delete	98
19.9.5 Auto-discard of expired IPMs	99
20 Message contents	99
20.1 Content	99
20.2 Content type	99
20.3 Content length	100
20.4 Encoded information types	100
21 Port realization	101
22 Conformance	101
22.1 Origination versus Reception	101
22.2 Statement requirements	101
22.3 Static requirements	101
22.4 Dynamic requirements	102
Annex A – Heading extensions	103
A.1 Incomplete Copy	103
A.2 Languages	103
A.3 Auto-submitted	103
Annex B – IPMS security extensions	104
B.1 Recipient Security Request	104
B.2 IPN Security Response	105
B.3 Security Diagnostic Code	106
B.4 Additional UA procedures	107
B.4.1 Originate IPM	107
B.4.2 Originate IPN	107
B.4.2.1 Message Submission	107
B.4.2.2 IPN Security Response procedures	108
B.4.2.2.1 Precedence of requests	108
B.4.2.2.2 Content-non-repudiation requested	108
B.4.2.2.3 Content-proof requested	109
B.4.2.2.4 Ipn-non-repudiation requested	110
B.4.2.2.5 Ipn-proof requested	110
B.5 Additional MS procedures	110
Annex C – Reference definition of Object Identifiers	111

	<i>Page</i>
Annex D – Reference definition of abstract information objects	116
Annex E – Reference definition of Extended Body Part Types	124
E.1 Equivalents of Basic Body Part Types.....	124
E.2 General Text.....	125
E.3 File Transfer.....	126
E.4 Voice.....	129
E.5 Report and Notification.....	129
E.6 Forwarded Content.....	130
Annex F – Reference definition of functional objects	132
Annex G – Reference definition of Abstract Service	133
Annex H – Reference definition of heading extensions	136
Annex I – Reference definition of Message Store attributes	137
Annex J – Reference definition of IPMS-MS auto-actions	149
Annex K – Reference definition of IPMS security extensions	153
Annex L – Reference definition of upper bounds.....	155
Annex M – Support of the Interpersonal Messaging Service	156
M.1 Support of recipient specifier components.....	156
M.2 Support of heading fields	157
M.3 Support of Body aspects	157
M.4 Support of IPMS Message Store.....	158
Annex N – Security model supplement for IPMS	159
N.1 Introduction.....	159
N.2 Security services	159
N.3 Supplements to clause 10.2: Security services.....	159
N.4 IPN Authentication	159
N.4.1 Proof of Notification.....	159
N.4.2 Proof of Content.....	159
N.5 Non-repudiation of IPM Responsibility.....	159
N.5.1 Non-repudiation of Notification	159
N.5.2 Non-repudiation of Content	159
Annex O – Differences between ISO/IEC 10021-7 and ITU-T Recommendation X.420.....	161
Annex P – Summary of changes to previous editions	162
P.1 Differences between CCITT Rec. X.420 (1984) and CCITT Rec. X.420 (1988)	162
P.2 Differences between CCITT Rec. X.420 (1988) and ISO/IEC 10021-7:1990	162
P.3 Differences between ISO/IEC 10021-7:1990 and CCITT Rec. X.420 (1992)	163
P.4 Differences between CCITT Rec. X.420 (1992) and ITU-T Rec. X.420 (1996) ISO/IEC 10021-7:1997	163
Annex Q – Index.....	164

AUSTRALIAN/NEW ZEALAND STANDARD

**Information technology—Text communication—
Message-oriented text interchange systems****Part 7:
Interpersonal messaging system****SECTION 1 – INTRODUCTION****1 Scope**

This Recommendation | International Standard defines **Interpersonal Messaging**, a form of Message Handling tailored for ordinary interpersonal business or private correspondence.

This Recommendation | International Standard is one of a series on Message Handling. ITU-T Rec. X.402 | ISO/IEC 10021-2 constitutes the introduction to the series and identifies the other documents in it.

The architectural basis and foundation for Message Handling are defined in still other Recommendations | International Standards. ITU-T Rec. X.402 | ISO/IEC 10021-2 identifies those documents as well.

This Recommendation | International Standard is structured as follows. Section 1 is this introduction. Section 2 defines the kinds of information objects exchanged in Interpersonal Messaging. Section 3 defines the associated abstract service. Section 4 specifies how it is provided. Annexes provide important supplemental information.

The requirements for conformance to this Recommendation | International Standard are given in clause 22.

2 Normative references

The following Recommendations and International Standards contain provisions which, through reference in this text, constitute provisions of this Recommendation | International Standard. At the time of publication, the editions indicated were valid. All Recommendations and Standards are subject to revision, and parties to agreements based on this Recommendation | International Standard are encouraged to investigate the possibility of applying the most recent edition of the Recommendations and Standards listed below. Members of ISO and IEC maintain registers of currently valid International Standards. The Telecommunication Standardization Bureau of the ITU maintains a list of currently valid ITU-T Recommendations.

2.1 Open Systems Interconnection

This Specification cites the following OSI specifications:

Identical Recommendations | International Standards

- ITU-T Recommendation X.227 (1995) | ISO/IEC 8650-1:1996, *Information technology – Open Systems Interconnection – Connection-oriented protocol for the Association Control Service Element: Protocol specification.*
- ITU-T Recommendation X.680 (1994) | ISO/IEC 8824-1:1995, *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation.*
- ITU-T Recommendation X.681 (1994) | ISO/IEC 8824-2:1995, *Information technology – Abstract Syntax Notation One (ASN.1): Information object specification.*
- ITU-T Recommendation X.682 (1994) | ISO/IEC 8824-3:1995, *Information technology – Abstract Syntax Notation One (ASN.1): Constraint specification.*
- ITU-T Recommendation X.683 (1994) | ISO/IEC 8824-4:1995, *Information technology – Abstract Syntax Notation One (ASN.1): Parameterization of ASN.1 specifications.*

- ITU-T Recommendation X.690 (1994) | ISO/IEC 8825-1:1995, *Information technology – ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)*.
- ITU-T Recommendation X.880 (1994) | ISO/IEC 13712-1:1995, *Information technology – Remote Operations: Concepts, model and notation*.

2.2 Message Handling Systems

This Specification cites the following Message Handling System specifications:

Identical Recommendations | International Standards

- ITU-T Recommendation X.402 (1995) | ISO/IEC 10021-2:1996, *Information technology – Message Handling Systems (MHS): Overall architecture*.
- ITU-T Recommendation X.411 (1995) | ISO/IEC 10021-4:1997, *Information technology – Message Handling Systems (MHS): Message transfer system: Abstract service definition and procedures*.
- ITU-T Recommendation X.413 (1995) | ISO/IEC 10021-5:1995, *Information technology – Message Handling Systems (MHS): Message store: Abstract service definition*.
- ITU-T Recommendation X.419 (1995) | ISO/IEC 10021-6:1996, *Information technology – Message Handling Systems (MHS): Protocol specifications*.

Paired Recommendations | International Standards equivalent in technical content

- ITU-T Recommendation F.400/X.400 (1996), *Message handling: System and service overview*.
ISO/IEC 10021-1:1997, *Information technology – Message Handling Systems (MHS) – Part 1: System and Service Overview*.

Additional references

- CCITT Recommendation X.408 (1988), *Message handling systems: Encoded information type conversion rules*.
- CCITT Recommendation X.420 (1984), *Message handling systems: Interpersonal messaging system*.

2.3 Directory Systems

This Specification cites the following Directory System specifications:

Identical Recommendations | International Standards

- ITU-T Recommendation X.501 (1993) | ISO/IEC 9594-2:1995, *Information technology – Open Systems Interconnection – The Directory: Models*.
- ITU-T Recommendation X.520 (1993) | ISO/IEC 9594-6:1995, *Information technology – Open Systems Interconnection – The Directory: Selected attribute types*.

2.4 Language Code

This Specification cites the following Language Code specification:

- ISO 639:1988, *Code for the representation of names of languages*.

2.5 Character Sets

This Specification cites the following Character Set specifications:

- ISO/IEC 2022:1994, *Information technology – Character code structure and extension techniques*.
- ISO 2375:1985, *Data processing – Procedure for registration of escape sequences*.
- ISO 8859-1:1987, *Information processing – 8-bit single-byte coded graphic character sets – Part 1: Latin alphabet No. 1*.
- CCITT Recommendation T.61 (1988), *Character repertoire and coded character sets for the international Teletex service*.

2.6 Telematic services

This Specification cites the following Telematic Service specifications:

- ITU-T Recommendation T.4 (1993), *Standardization of group 3 facsimile apparatus for document transmission*.
- ITU-T Recommendation T.30 (1993), *Procedures for document facsimile transmission in the general switched telephone network*.
- CCITT Recommendation T.100 (1984), *International information exchange for interactive videotex*.
- ITU-T Recommendation T.101 (1994), *International interworking for videotex services*.
- CCITT Recommendation T.330 (1988), *Telematic access to interpersonal message system*.

2.7 File Transfer

This Specification cites the following File Transfer specifications:

- ISO 8571-1:1988, *Information processing systems – Open Systems Interconnection – File Transfer, Access and Management – Part 1: General introduction*.
- ISO 8571-2:1988, *Information processing systems – Open Systems Interconnection – File Transfer, Access and Management – Part 2: Virtual Filestore Definition*.
- ISO 8571-2:1988/Amd.1:1992, *Information processing systems – Open Systems Interconnection – File Transfer, Access and Management – Part 2: Virtual Filestore Definition – Amendment 1: Filestore Management*.
- ISO 8571-4:1988, *Information processing systems – Open Systems Interconnection – File Transfer, Access and Management – Part 4: File Protocol Specification*.
- ISO 8571-4:1988/Amd.1:1992, *Information processing systems – Open Systems Interconnection – File Transfer, Access and Management – Part 4: File Protocol Specification – Amendment 1: Filestore Management*.

2.8 Open Document Architecture

This Specification cites the following Open Document Architecture specifications:

- ITU-T Recommendation T.415 (1993) | ISO/IEC 8613-5:1994, *Information technology – Open Document Architecture (ODA) and interchange format: Open Document Interchange Format*.

2.9 Digital Encoding of Sound

This Specification cites the following specifications on the Digital Encoding of Sound:

- CCITT Recommendation G.711 (1988), *Pulse Code Modulation (PCM) of voice frequencies*.
- CCITT Recommendation G.726 (1990), *40, 32, 24, 16 kbit/s Adaptive Differential Pulse Code Modulation (ADPCM)*.
- CCITT Recommendation G.728 (1992), *Coding of speech at 16 kbit/s using low-delay code excited linear prediction*.
- IEC 908:1987, *Compact disc digital audio system*.

3 Definitions

For the purposes of this Specification, the definitions given in ITU-T Rec. X.402 | ISO/IEC 10021-2 apply.

4 Abbreviations

For the purposes of this Specification, the abbreviations given in ITU-T Rec. X.402 | ISO/IEC 10021-2 apply.

5 Conventions

This Specification uses the descriptive conventions identified below.

5.1 ASN.1

This Specification uses, for the indicated purposes, the following ASN.1-based descriptive conventions:

- a) To define the information objects of Interpersonal Messaging, and other data types and values of all kinds, ASN.1 itself. ASN.1 is defined in ITU-T Rec. X.680 | ISO/IEC 8824-1, ITU-T Rec. X.681 | ISO/IEC 8824-2, ITU-T Rec. X.682 | ISO/IEC 8824-3 and ITU-T Rec. X.683 | ISO/IEC 8824-4.
- b) To define the functional objects of Interpersonal Messaging, the MHS-OBJECT information object class of ITU-T Rec. X.411 | ISO/IEC 10021-4.
- c) To define the abstract service of Interpersonal Messaging, the PORT and ABSTRACT-OPERATION and ABSTRACT-ERROR information object class of ITU-T Rec. X.411 | ISO/IEC 10021-4 and the CONTRACT information object class of ITU-T Rec. X.880 | ISO/IEC 13712-1.
- d) To define the *IPMS extensions*, the IPMS-EXTENSION information object class of 7.2.17.
- e) To define *extended body part types*, the EXTENDED-BODY-PART-TYPE information object class of 7.3.1.
- f) To define IPMS-MS attributes, the ATTRIBUTE information object class of ITU-T Rec. X.413 | ISO/IEC 10021-5.

The abstract-syntax defined in this Specification may be mapped to that used in previous editions as follows. All ASN.1 definitions of object sets and Enumerated types which contain the ASN.1 extensions marker ("...") are treated as if any extension additions following the marker are absent. For ASN.1 definitions where the extension marker is not used, the ASN.1 comment "-- 1994 extension --" has a similar interpretation. See 5.7 of ITU-T Rec. X.413 | ISO/IEC 10021-5. The effect of this is that certain attribute-types, matching-rules, and auto-actions are not standardized for use in 1988 Application Contexts.

The various uses of the ASN.1 notation are summarized in Table 1. With the two exceptions evident from the table, whenever ASN.1 is employed, it appears both in the body of this Specification to aid the exposition, and again, largely redundantly, in an annex for reference.

Table 1 – Uses of the ASN.1 notation

Subject matter	Exposition	Reference
Object identifiers	–	Annex C
Abstract information objects	Section 2	Annex D
Extended body part types	Clauses 7.3, 7.4	Annex E
Functional objects	Clauses 10, 11, 16	Annex F
Abstract service	Clauses 12-13	Annex G
Message store attributes	Clause 19	Annex I
Message store auto-actions	Clause 19	Annex J
Heading extensions	Annex A	Annex H
Security extensions	Annex B	Annex K
Upper bounds	–	Annex L

If differences are found between the ASN.1 used in the exposition and that supplied for reference, a specification error is indicated.

Except for Annex J, ASN.1 tags are implicit throughout the ASN.1 module the annex defines; the module is definitive in that respect.

NOTES

1 The use of ASN.1 to describe a class or piece of information does not in itself imply that that information is transported between open systems. The fact that the information, by virtue of its description in ASN.1 and of ASN.1's Basic Encoding Rules, has a concrete transfer syntax may be immaterial. Information actually conveyed between systems is designated as such by its inclusion in an application protocol.

2 The use of the ABSTRACT-OPERATION and -ERROR information object classes, derived from the correspondingly named information object classes of Remote Operations, does not imply that the abstract operations and errors are invoked and reported across the boundary between open systems. The fact that the abstract operations and errors, by virtue of their description using these information object classes and with minimal additional specification, actually could be invoked via ROS is immaterial in the present context.

5.2 Grade

This Specification uses the concept of grade as developed in ITU-T Rec. X.402 | ISO/IEC 10021-2.

5.3 Terms

Throughout this Specification, terms are rendered in **bold** when defined, in *italic* when referenced prior to their definitions, without emphasis upon all other occasions.

Terms that are proper nouns are capitalized, generic terms are not.

5.4 Conventions for attribute-types used in Table 5

This Specification uses the conventions listed below in its definition of the attribute-types for the IPMS-MS abstract-service.

For the column headed 'Single/Multi-valued', the following values can occur:

S single-valued

M multi-valued

For the column headed 'Source', the following values can occur:

IPM Originate-IPM, Receive-IPM abstract-operations;

Mod Modify abstract-operation;

MS IPMS Message Store;

NRN Originate-NRN, Receive-NRN abstract-operations;

ON Originate other-notifications, Receive other-notifications;

RN Originate-RN, Receive-RN abstract-operations.

SECTION 2 – ABSTRACT INFORMATION OBJECTS

6 Overview

This section abstractly describes the information objects that users exchange in Interpersonal Messaging. They are of two kinds, *Interpersonal Messages (IPMs)* and *Interpersonal Notifications (IPNs)*. One of the latter acknowledges a user's receipt of one of the former.

```
InformationObject ::= CHOICE {
    ipm [0] IPM,
    ipn [1] IPN}
```

This section covers the following topics:

- a) Interpersonal messages;
- b) Interpersonal notifications.

NOTES

1 The use, throughout this section, of words such as “originator” and “recipient” anticipates the fact that *IPMs* and *IPNs* are conveyed between users as the contents of messages (see clause 20). These words, therefore, refer to the roles users and DLs play in such transmittals.

2 An *IPM* may appear (see 7.4.7) in the *Body* of another *IPM* which itself is conveyed as the content of a message. The words “originator” and “recipient” shall be understood in the context of an *IPM*'s conveyance as the (entire) content of a message, not as a component of the *Body* of another *IPM* so conveyed.

3 An *IPM* or *IPN* makes various assertions about its own transmittal (e.g. who originates the message containing it). Furthermore, an *IPN* makes assertions about the transmittal of the *IPM* to which it responds. All of these assertions are unverified.

7 Interpersonal messages

An **Interpersonal Message (IPM)** is a member of the primary class of information object conveyed between users in Interpersonal Messaging.

```
IPM ::= SEQUENCE {
    heading Heading,
    body Body}
```

It has the following components:

- Heading:** A Set of **heading fields** (or **fields**), each an information item that gives a characteristic of the IPM (e.g. its importance).
- Body:** A Sequence of **body parts**, each an information object that the IPM is intended to convey between users (e.g. a document).

```
Body ::= SEQUENCE OF BodyPart
```

The structure of an IPM is depicted in Figure 1.

This clause defines and describes the most prominent Heading field component types and the defined Heading fields and body part types.

NOTE – An IPM may be likened to a business memo. In fact, the terms “Heading” and “Body” appeal to that analogy.

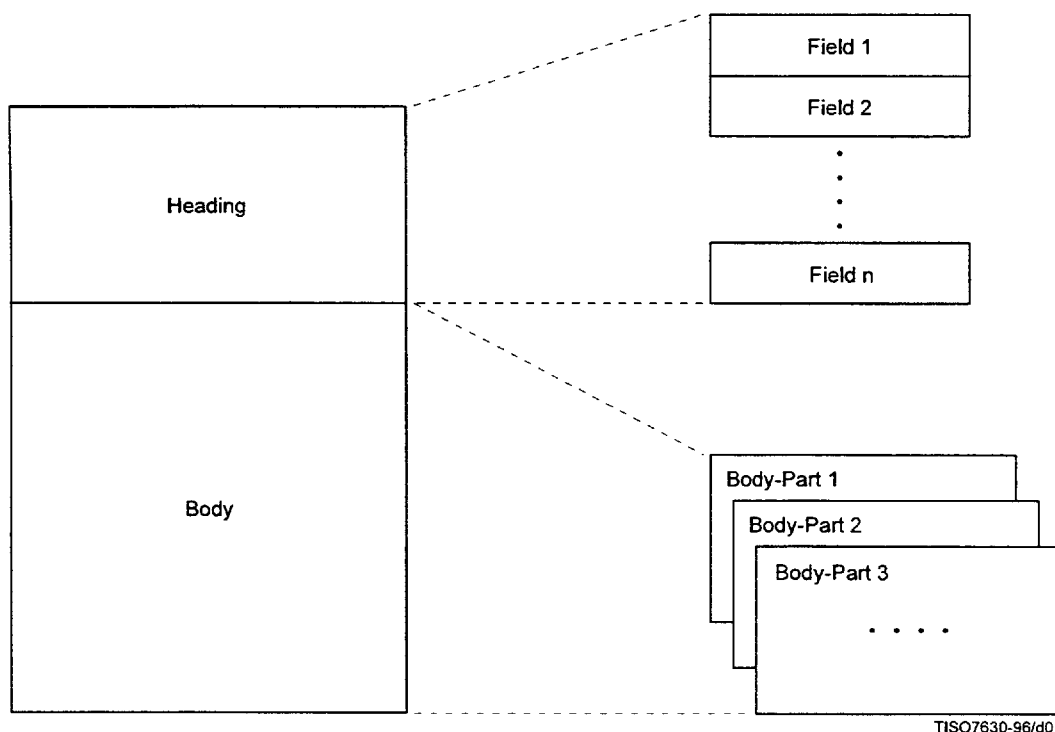


Figure 1 – An Interpersonal Message

7.1 Heading field component types

Information items of several kinds appear throughout the Heading. These Heading field component types -- *IPM identifier*, *recipient specifier*, *OR-descriptor* and *IPMS extension* -- are defined and described below.

7.1.1 IPM identifier

An **IPM identifier** is an information item that unambiguously and uniquely identifies an IPM, distinguishing it from all other IPMs ever conveyed by any user.

```
IPMIdentifier ::= [APPLICATION 11] SET {
    user ORName OPTIONAL,
    user-relative-identifier LocalIPMIdentifier}
```

An IPM identifier has the following components:

- a) **User (O)**: Identifies the user who originates the IPM. One of the user's OR-names. This component's omission is discouraged.
- b) **User-relative-identifier (M)**: Uniquely and unambiguously identifies the IPM, distinguishing it from all other IPMs that the user who is identified by the User component originates. A Printable String[of from zero to a prescribed number of characters (see Annex L)]. A length of zero is discouraged.

LocalIPMIdentifier ::= PrintableString
(SIZE (0..ub-local-ipm-identifier))

NOTE – The "11" in IPMIdentifier is the only ASN.1 application-wide tag this Specification assigns.

7.1.2 Recipient specifier

A **recipient specifier** is an information item that identifies an (intended) recipient of an IPM and that may make certain requests of him.

RecipientSpecifier ::= SET {
recipient [0] ORDescriptor,
notification-requests [1] NotificationRequests DEFAULT {},
reply-requested [2] BOOLEAN DEFAULT FALSE,
recipient-extensions [3] RecipientExtensionsField OPTIONAL}

A recipient specifier has the following components:

- a) **Recipient (M)**: Identifies the recipient in question. An *OR-descriptor*.
 If the *Notification-requests* or *Reply-requested* component makes a request of the recipient, the *Formal-name* component of the *OR-descriptor* above shall be present.
- b) **Notification-requests (D no values)**: May make certain requests of the recipient denoted by the Recipient component.

NotificationRequests ::= BIT STRING {
rn (0),
nrn (1),
ipm-return (2)}

This component may assume any of the following values simultaneously, except that the value *rn* shall not be selected unless the value *nrn* is selected:

- i) *rn*: A *receipt notification* is requested in the circumstances prescribed in clause 8.
- ii) *nrn*: A *non-receipt notification* is requested in the circumstances prescribed in clause 8.
- iii) *ipm-return*: It is requested that the IPM be returned in any *non-receipt notification*.
- c) **Reply-requested (D false)**: Indicates whether a reply is requested of the recipient denoted by the Recipient component. A Boolean.

A **reply** is one IPM sent in response to another. A user may reply to an IPM even though no reply is requested of him and, indeed, even if he is not among the IPM's intended recipients. Furthermore, a user of whom a reply is requested may refrain from replying.

- d) **Recipient-extensions (O)** contains extensions to the recipient specifier subfield.

RecipientExtensionsField ::= SET OF IPMSExtension

Recipient extensions for IPMS Security are defined in Annex B. There are no other recipient extensions defined in this Specification.

7.1.3 OR-descriptor

An **OR-descriptor** is an information item that identifies a user or DL.

ORDescriptor ::= SET {
formal-name ORName OPTIONAL,
free-form-name [0] FreeFormName OPTIONAL,
telephone-number [1] TelephoneNumber OPTIONAL}

COPYRIGHT

An OR-descriptor has the following components:

- a) **Formal-name (C)**: Identifies the user or DL in question. One of its OR-names.
This conditional component shall be present if (but not only if) one or more of the following criteria are satisfied:
- i) The *Free-form-name* component is absent.
 - ii) The OR-descriptor appears in the *Reply Recipients* heading field.
 - iii) The OR-descriptor is the Recipient component of a recipient specifier and the conditions stated in item a of 7.1.2 are satisfied.

- b) **Free-form-name (O)**: Identifies the user or DL in question. A Teletex String[of from zero to a prescribed number of characters (see annex L)], chosen from the graphic subset of the Teletex String character set. A length of zero is discouraged.

FreeFormName ::= TeletexString (SIZE (0..ub-free-form-name))

- c) **Telephone-number (O)**: Provides the telephone number of the user or DL in question. A Printable String[of from zero to a prescribed number of characters (see annex L)], chosen from the graphical subset of the Printable String character set. A length of zero is discouraged.

PhoneNumber ::= PrintableString (SIZE (0..ub-telephone-number))

NOTE – One or more OR-descriptors may appear in each of the following heading fields: Originator, Authorizing Users, Primary Recipients, Copy Recipients, Blind Copy Recipients, and Reply Recipients. In addition, an OR-descriptor may appear in the following notification fields (see clause 8): IPN Originator and IPM Intended Recipient.

7.1.4 IPMS extension

An **IPMS extension** (or **extension**) is an information item which enables extensions to be defined.

IPMSExtension ::= SEQUENCE {
 type IPMS-EXTENSION.&id,
 value IPMS-EXTENSION.&Type DEFAULT NULL:NULL }

Each extension has the following components:

- a) **Type (M)**: Identifies the semantics and restricts the abstract syntax of the *Value* component. An Object Identifier.
- b) **Value (D null)**: An information item whose abstract syntax is restricted only by the *Type* component. An Any.

The *Type* components of all the extensions in the Extensions field shall differ. Not every defined extension need appear in the field.

Every extension is defined by means of the following information object class.

IPMS-EXTENSION ::= CLASS {
 &id OBJECT IDENTIFIER UNIQUE,
 &Type DEFAULT NULL }
WITH SYNTAX { [VALUE &Type ,] IDENTIFIED BY &id }

The *&id* value field is the identifier field for the class. It distinguishes an IPMS Extension from all other instances of the class.

The *&Type* field defines the data type to which every value of this instance of IPMS Extension shall conform.

NOTE – Whenever an IPMS extension is defined, it should also be considered whether new MS attributes should also be defined (see 19.6), and whether the UA operation needs to be modified (see clause 18). As any extension may be ignored, privately-defined extensions are not able to mandate support of the new MS attributes or UA operation.

7.2 Heading fields

The fields that may appear in the Heading of an IPM are defined and described below.

Heading ::= SET {
 this-IPM **ThisIPMField,**
 originator **[0] OriginatorField OPTIONAL,**
 authorizing-users **[1] AuthorizingUsersField OPTIONAL,**
 primary-recipients **[2] PrimaryRecipientsField DEFAULT {},**

COPYRIGHT

copy-recipients	[3]	CopyRecipientsField DEFAULT {},
blind-copy-recipients	[4]	BlindCopyRecipientsField OPTIONAL,
replied-to-IPM	[5]	RepliedToIPMField OPTIONAL,
obsoleted-IPMs	[6]	ObsoletedIPMsField DEFAULT {},
related-IPMs	[7]	RelatedIPMsField DEFAULT {},
subject	[8]	EXPLICIT SubjectField OPTIONAL,
expiry-time	[9]	ExpiryTimeField OPTIONAL,
reply-time	[10]	ReplyTimeField OPTIONAL,
reply-recipients	[11]	ReplyRecipientsField OPTIONAL,
importance	[12]	ImportanceField DEFAULT normal,
sensitivity	[13]	SensitivityField OPTIONAL,
auto-forwarded	[14]	AutoForwardedField DEFAULT FALSE,
extensions	[15]	ExtensionsField DEFAULT {}

Some fields have components and thus are composite, rather than indivisible. A field component is called a **sub-field**.

7.2.1 This IPM

The **This IPM** heading field (M) identifies the IPM. It comprises an IPM identifier.

ThisIPMField ::= IPMIdentifier

7.2.2 Originator

The **Originator** heading field (O) identifies the IPM's originator. It comprises an OR-descriptor.

OriginatorField ::= ORDescriptor

7.2.3 Authorizing Users

The **Authorizing Users** heading field (C) identifies the zero or more users who are the IPM's *authorizing users*. It comprises a Sequence of sub-fields, each an OR-descriptor, one for each such user.

AuthorizingUsersField ::= SEQUENCE OF AuthorizingUsersSubfield

AuthorizingUsersSubfield ::= ORDescriptor

An **authorizing user** is a user who, either individually or in concert with others, authorizes the origination of an IPM. The word "authorizes" above is not precisely defined by this Specification; it is given meaning by users.

This conditional field shall be present if the authorizing users are other than the IPM's originator alone.

NOTE – Suppose, e.g. that a manager instructs his secretary to originate an IPM on his behalf. In this case, the secretary, the IPM's originator, might consider the manager the authorizing user.

7.2.4 Primary Recipients

The **Primary Recipients** heading field [D no subfields (i.e. elements)] identifies the zero or more users and DLs who are the "primary recipients" of the IPM. It also identifies the responses the authorizing users ask of each of those users and of each member of those DLs. It comprises a Sequence of sub-fields, each a recipient specifier, one for each primary recipient.

PrimaryRecipientsField ::= SEQUENCE OF PrimaryRecipientsSubfield

PrimaryRecipientsSubfield ::= RecipientSpecifier

The phrase "primary recipients" above is not precisely defined by this Specification; it is given meaning by users.

NOTE – The primary recipients, e.g. might be those users and those DLs whose members are expected to act upon the IPM.

7.2.5 Copy Recipients

The **Copy Recipients** heading field [D no subfields (i.e. elements)] identifies the zero or more users and DLs who are the "copy recipients" of the IPM. It also identifies the responses the authorizing users ask of each of those users and of each member of those DLs. It comprises a Sequence of sub-fields, each a recipient specifier, one for each copy recipient.

CopyRecipientsField ::= SEQUENCE OF CopyRecipientsSubfield

CopyRecipientsSubfield ::= RecipientSpecifier

The phrase “copy recipients” above is not precisely defined by this Specification; it is given meaning by users.

NOTE – The copy recipients, e.g. might be those users to whom, and those DLs to whose members the IPM is conveyed for information.

7.2.6 Blind Copy Recipients

The **Blind Copy Recipients** heading field (C) identifies zero or more users and DLs who are intended *blind* copy “recipients” of the IPM. It also identifies the responses the authorizing users ask of each of those users and of each member of those DLs. It comprises a Sequence of sub-fields, each a recipient specifier, one for each *blind* copy recipient.

BlindCopyRecipientsField ::= SEQUENCE OF **BlindCopyRecipientsSubfield**

BlindCopyRecipientsSubfield ::= **RecipientSpecifier**

The phrase “copy recipients” above has the same meaning as in 7.2.5. A **blind** copy recipient is one whose role as such is disclosed to neither primary nor copy recipients.

In the instance of an IPM intended for a blind copy recipient, this conditional field shall be present and identify that user or DL. Whether it shall also identify the other blind copy recipients is a local matter. In the instance of the IPM intended for a primary or copy recipient, the field shall be absent or identify no users or DLs.

7.2.7 Replied-to IPM

The **Replied-to IPM** heading field (C) identifies the IPM to which the present IPM is a reply. It comprises an IPM identifier.

RepliedToIPMField ::= **IPMIdentifier**

This conditional field shall be present if, and only if, the IPM is a reply.

NOTE – In the context of *forwarding*, care should be taken to distinguish between the *forwarding IPM* and the *forwarded IPM*. This field should identify whichever of these two IPMs to which the reply responds.

7.2.8 Obsolete IPMs

The **Obsolete IPMs** heading field [D no subfields (i.e. elements)] identifies zero or more IPMs that the authorizing users of the present IPM consider it to obsolete. It comprises a Sequence of sub-fields, each an IPM identifier, one for each IPM.

ObsoleteIPMsField ::= SEQUENCE OF **ObsoleteIPMsSubfield**

ObsoleteIPMsSubfield ::= **IPMIdentifier**

NOTE – In the context of *forwarding*, care should be taken to distinguish between the *forwarding IPM* and the *forwarded IPM*. This field should identify whichever of these two IPMs the present IPM obsoletes.

7.2.9 Related IPMs

The **Related IPMs** heading field [D no subfields (i.e. elements)] identifies zero or more IPMs that the authorizing users of the present IPM consider related to it. It comprises a Sequence of sub-fields, each an IPM identifier, one for each IPM.

RelatedIPMsField ::= SEQUENCE OF **RelatedIPMsSubfield**

RelatedIPMsSubfield ::= **IPMIdentifier**

The word “related” above is not precisely defined by this Specification; it is given meaning by users.

NOTES

1 A related IPM, e.g. might be one discussed in the Body of the present IPM.

2 In the context of *forwarding*, care should be taken to distinguish between the *forwarding IPM* and the *forwarded IPM*. This field should identify whichever of these two IPMs is related to the present IPM.

7.2.10 Subject

The **Subject** heading field (O) identifies the subject of the IPM. It comprises a Teletex String[of from zero to a prescribed number of characters (see Annex L)], chosen from the graphic subset of the Teletex String character set. A length of zero is discouraged.

SubjectField ::= TeletexString (SIZE (0..ub-subject-field))

COPYRIGHT

7.2.11 Expiry Time

The **Expiry Time** heading field (O) identifies when the authorizing users consider the IPM to lose its validity. It comprises a date and time.

ExpiryTimeField ::= Time

7.2.12 Reply Time

The **Reply Time** heading field (O) identifies by when the authorizing users request (but do not demand) that any replies to the present IPM be originated. It comprises a date and time.

ReplyTimeField ::= Time

7.2.13 Reply Recipients

The **Reply Recipients** heading field (C) identifies zero or more users and DLs whom the authorizing users request (but do not demand) be among the recipients of any replies to the present IPM. It comprises a Sequence of sub-fields, each an OR-descriptor, one for each user or DL.

ReplyRecipientsField ::= SEQUENCE OF ReplyRecipientsSubfield

ReplyRecipientsSubfield ::= ORDescriptor

This conditional field shall be present if the desired reply recipients are other than the originator of the present IPM alone.

NOTE – If this field is present and identifies several users and DLs, the originator may include himself among them. If he elects not to do so, he will not be considered among the desired reply recipients.

7.2.14 Importance

The **Importance** heading field (D *normal*) identifies the importance that the authorizing users attach to the IPM. It may assume any one of the following values: *low*, *normal*, or *high*.

ImportanceField ::= ENUMERATED {
 low **(0),**
 normal **(1),**
 high **(2)}**

The values above are not defined by this Specification; they are given meaning by users.

7.2.15 Sensitivity

The **Sensitivity** heading field (C) identifies the sensitivity that the authorizing users attribute to the IPM.

SensitivityField ::= ENUMERATED {
 personal **(1),**
 private **(2),**
 company-confidential **(3)}**

This field may assume any one of the following values:

- a) *personal*: The IPM is conveyed to its intended recipients as individuals, rather than in their professional capacities.
- b) *private*: The IPM should be conveyed to no one other than its intended recipients.
- c) *company-confidential*: The IPM contains information that should be handled according to company-specific procedures.

This conditional field shall be present if, and only if, the IPM is sensitive.

7.2.16 Auto-forwarded

The **Auto-forwarded** heading field (D *false*) indicates whether the IPM is the result of *auto-forwarding*. It is a Boolean.

AutoForwardedField ::= BOOLEAN

7.2.17 Extensions

The **Extensions** heading field [D no *extensions* (i.e. members)] conveys information accommodated by no other heading field. It comprises a Set of zero or more IPMS extensions, each conveying one such information item.

ExtensionsField ::= SET OF IPMSExtension

All heading extensions defined in this Specification are contained in Annex A. An extension whose Type component is not understood may be ignored.

NOTE – Future addenda or versions of this Specification may define additional extensions. Furthermore, future addenda and versions are likely to add information to the Heading only by means of this field.

7.3 Body Parts

The **IPM Body Part Table** information object set has as its members the types of body part that may appear in the Body of an IPM:

IPMBodyPartTable EXTENDED-BODY-PART-TYPE ::= {
StandardBodyParts |
ApplicationSpecificBodyParts }

The **Standard Body Parts** information object set has as its members the types of body part defined in this Specification:

StandardBodyParts EXTENDED-BODY-PART-TYPE ::= {
ia5-text-body-part |
g3-facsimile-body-part |
g4-class1-body-part |
teletex-body-part |
videotex-body-part |
encrypted-body-part |
message-body-part |
mixed-mode-body-part |
bilaterally-defined-body-part |
nationally-defined-body-part |
general-text-body-part |
file-transfer-body-part |
voice-body-part |
report-body-part |
notification-body-part |
Content-Body-Parts,
... }

The **Application Specific Body Parts** information object set has as its members the types of body part defined in other Specifications, and defined for proprietary or private use:

ApplicationSpecificBodyParts EXTENDED-BODY-PART-TYPE ::= {
-- any body part defined in other Specifications, or for proprietary or private use --
... }

Some types of body part contain a single component, the *Data* component. Other types of body part contain two components, *Parameters* and *Data*. The **Parameters** component, if defined, comprises information items that describe the information object the body part represents, and typically contains format and control parameters. The **Data** component is the information object itself.

7.3.1 Extended Body Part

Every body part type defined in this Specification is defined as an instance of the **Extended Body Part Type** information object class. Every body part type defined elsewhere shall be defined in the same way.

EXTENDED-BODY-PART-TYPE ::= CLASS {
¶meters TYPE-IDENTIFIER OPTIONAL,
&data TYPE-IDENTIFIER }
WITH SYNTAX { [PARAMETERS ¶meters,] DATA &data }

An instance of the Extended Body Part Type information object class defines, by means of its ¶meters field, the type of the data value that is represented by the Parameters component of such a body part, and the Object Identifier that identifies this Parameters component. The presence of the ¶meters field implies the presence of the Parameters component in every instance of that specific extended body part type; its omission implies the absence of the Parameters component in every instance.

An instance of the Extended Body Part Type information object class also defines, by means of its &data field, the type of the data value that is represented by the Data component of such a body part, and the Object Identifier that identifies this Data component. The Object Identifier identifies the encoding rules for the body part. Those body parts whose types are defined in this Specification shall be encoded using ASN.1's basic encoding rules.

Specific Extended body part types may be defined by users to convey any type of information object. The definitions in 7.4.11, 7.4.12, 7.4.14 and 7.4.15 give examples where the information object is an ASN.1 Type, while the examples below show definitions for information objects not defined by ASN.1. User-defined Extended body part types are appropriate where the recipient's UA will be configured to invoke a process to render that body part, but where the requirement is to move information into the recipient's filestore the File Transfer body part (see 7.4.12) may be more appropriate.

Examples

An organization has two products for which it wishes to define Extended body parts. Its 'Simple Spreadsheet' uses a single file of octets for its data, while its 'Whizzbang Wordprocessor' always uses two related data files of octets (e.g. one for format proforma, and one for the text using these proforma). The organization has obtained a branch of the Object Identifier tree from its national registration authority, called here *local-object-identifier*.

The following definition is an Extended body part for the 'Simple Spreadsheet' data file, and the same object identifier ({ local-object-identifier 1 }) can be used for its Encoded Information Type:

```
simple-spreadsheet-body-part EXTENDED-BODY-PART-TYPE ::= {
  DATA    { OCTET STRING IDENTIFIED BY { local-object-identifier 1 } } }
```

The following definition is an Extended body part for the 'Whizzbang Wordprocessor' data files, with the format proforma file being placed in the Parameters component and the text file in the Data component. The same object identifier as used for the Data component ({ local-object-identifier 3 }) can be used for its Encoded Information Type:

```
whizzbang-wordprocessor-body-part EXTENDED-BODY-PART-TYPE ::= {
  PARAMETERS { OCTET STRING IDENTIFIED BY { local-object-identifier 2 } },
  DATA      { OCTET STRING IDENTIFIED BY { local-object-identifier 3 } } }
```

NOTES

1 This body part type enables the exchange of information objects of all kinds, each kind unambiguously identified. This unambiguous identification relies upon the use of Object Identifiers. Object Identifiers are easily obtained, e.g. by national bodies and private organizations. It is desirable that the identification is also unique, and anyone defining a data format is encouraged to define and publicize an Extended body part for that format in order to remove the likely non-uniqueness if such definition is left to the users of that data format.

2 If an Extended body part has a Parameters component, its Object Identifier is allocated at the same time and by the same naming authority as that for the Data component, and must be different to it.

3 When a new Extended body part type and a new Encoded Information Type are defined and have a one to one relationship, then the same Object Identifier may be used for the data component and the Encoded Information Type.

4 Like body parts of other types, an Extended body part may be subjected to conversion. However, specification of the conversion algorithms may be outside the scope of Recommendation X.408.

7.3.2 Body Part Encoding

Each body part present in the Body of an IPM is represented as follows:

```
BodyPart ::= CHOICE {
  basic    CHOICE {
    ia5-text      [0] IA5TextBodyPart,
    g3-facsimile  [3] G3FacsimileBodyPart,
    g4-class1     [4] G4Class1BodyPart,
    teletex       [5] TeletexBodyPart,
    videotex      [6] VideotexBodyPart,
    encrypted     [8] EncryptedBodyPart,
    message       [9] MessageBodyPart,
    mixed-mode    [11] MixedModeBodyPart,
    bilaterally-defined [14] BilaterallyDefinedBodyPart,
    nationally-defined [7] NationallyDefinedBodyPart },
  extended  [15] ExtendedBodyPart }
```

All body part types are divided into two classes as follows:

- a) **basic**: Denoted by an integer (an ASN.1 context-specific tag).

All basic body part types are defined in this Specification. Each basic body part type also has an equivalent extended body part definition.

- b) **extended**: Denoted by an Object Identifier.

Some extended body part types are defined in this Specification. Some are defined in other Recommendations | International Standards (e.g. the ODA body part defined in ITU-T Rec. T.411 | ISO/IEC 8613-1). Others may be defined by users. The extended body part type enables the exchange of information objects of any kind, each kind unambiguously and uniquely identified.

An instance of the extended body part contains an information object whose semantics and abstract syntax are denoted by the Object Identifier which the body part carries. It has Parameters and Data components.

```
ExtendedBodyPart{EXTENDED-BODY-PART-TYPE:IPMBodyPartTable} ::= SEQUENCE {
  parameters [0] INSTANCE OF TYPE-IDENTIFIER OPTIONAL,
  data        INSTANCE OF TYPE-IDENTIFIER }
  (CONSTRAINED BY {-- must correspond to the &parameters field and &data field
    -- of a member of -- IPMBodyPartTable})
```

The Parameters and Data components correspond to fields of the Extended Body Part Type information object class (see 7.3.1). Hence each component is defined as an instance of the Type-Identifier information object class (see Annex A of ITU-T Rec. X.681 | ISO/IEC 8824-2).

NOTES

1 In Recommendation X.420 (1984), context-specific tags 1 and 10 denote Telex and Simple Formattable Document body parts, respectively, which are no longer defined. In ISO DP 9065, context-specific tags 12 and 13 denote ODA and ISO 6937 Text body parts, respectively, which are no longer defined. In Recommendation X.420 (1984), CCITT Rec. X.420 (1988), and ISO/IEC 10021-7:1990, context-specific tag 2 denotes the Voice basic body part which is no longer defined. These tags, therefore, are avoided in BodyPart.

2 Under some circumstances, an IPM may be subjected to conversion while in transit between users. Such a transmittal event may alter a body part's type.

- 3 The basic body part types exist for purely historical reasons, predating the extended body part type.

4 In editions of this Specification published before 1994, the Parameters and Data components of extended body parts were defined as External. When the single-ASN1-type encoding alternative of the External is used, the value of the encoding of an External is identical with that of Type-Identifier. However, to accommodate the case where the octet-aligned encoding alternative is used, an associated type for the representation of the extended body part may be used on origination, and should be supported on reception. This associated type is assumed to be defined in an environment of EXPLICIT TAGS:

```
SEQUENCE {
  parameters [0] IMPLICIT SEQUENCE {
    direct-reference EXTENDED-BODY-PART-TYPE.&parameters.&id,
    encoding CHOICE {
      single-ASN1-type [0] EXTENDED-BODY-PART-TYPE.&parameters.&Type,
      octet-aligned      [1] IMPLICIT OCTET STRING } } OPTIONAL,
  data [UNIVERSAL 8] IMPLICIT SEQUENCE {
    direct-reference EXTENDED-BODY-PART-TYPE.&data.&id,
    encoding CHOICE {
      single-ASN1-type [0] EXTENDED-BODY-PART-TYPE.&data.&Type,
      octet-aligned      [1] IMPLICIT OCTET STRING } } }
```

If the Parameters or Data component is defined as an Octet String, or comprises octet-aligned data not defined using ASN.1, its encoding may occupy either the single-ASN1-type alternative (as an explicitly tagged Octet String) or the octet-aligned alternative (as an implicitly tagged Octet String). Otherwise, the encoding should occupy the single-ASN1-type alternative.

7.4 Standard Body Part Types

The standard body part types defined in this Specification are enumerated below.

7.4.1 IA5 Text

An IA5 Text body part represents IA5 text. It has Parameters and Data components.

```
IA5TextBodyPart ::= SEQUENCE {
    parameters  IA5TextParameters,
    data        IA5TextData}

ia5-text-body-part EXTENDED-BODY-PART-TYPE ::= {
    PARAMETERS  {IA5TextParameters IDENTIFIED BY id-ep-ia5-text},
    DATA        {IA5TextData IDENTIFIED BY id-et-ia5-text} }

IA5TextParameters ::= SET {
    repertoire [0] Repertoire DEFAULT ia5}

IA5TextData ::= IA5String
```

The Parameters component comprises the following parameters:

- **Repertoire (D IA5):** Identifies the character set to which the Data component is constrained.

```
Repertoire ::= ENUMERATED {
    ita2(2),
    ia5(5)}
```

This parameter may assume any one of the following values:

- i) *ITA2*: The Data component shall be limited to the ITA2 (i.e. Telex) character set.
- ii) *IA5*: The Data component may draw upon the full IA5 character set.

The Data component is the text, an IA5 String. It may contain lines of any length. Whenever the component is rendered (e.g. displayed to or printed for a user), all (rather than only a part) of the text must be communicated (e.g. lines may be folded but shall not be truncated).

NOTE – Many terminals have a maximum line length of 80 characters. Therefore, lines that do not exceed that length are most likely to be satisfactorily rendered (e.g. are most likely to avoid being folded).

7.4.2 G3 Facsimile

A G3 Facsimile body part represents Group 3 facsimile images. It has Parameters and Data components.

```
G3FacsimileBodyPart ::= SEQUENCE {
    parameters  G3FacsimileParameters,
    data        G3FacsimileData}

g3-facsimile-body-part EXTENDED-BODY-PART-TYPE ::= {
    PARAMETERS {G3FacsimileParameters IDENTIFIED BY id-ep-g3-facsimile},
    DATA        {G3FacsimileData IDENTIFIED BY id-et-g3-facsimile} }

G3FacsimileParameters ::= SET {
    number-of-pages      [0] INTEGER OPTIONAL,
    non-basic-parameters [1] G3FacsimileNonBasicParameters OPTIONAL}

G3FacsimileData ::= SEQUENCE OF BIT STRING
```

The Parameters component comprises the following parameters:

- a) **Number-of-pages (O):** Identifies the number of pages of Group 3 facsimile data present in the Data component. A non-negative Integer.
- b) **Non-basic-parameters (C):** Identifies the non-basic parameters (NBPs) for Group 3 facsimile that characterize the Data component. A G3 NBPs descriptor.

This conditional parameter may be absent if the data component is of basic G3 Facsimile type. If the data component is of a non-basic type, it shall be present if (but not only if) the Body contains two or more G3 Facsimile body parts. The absence of this parameter when the data component is of a non-basic type is discouraged.

NOTE 1 – Its absence in these conditions provides compatibility with Recommendation X.420 (1984).

The Data component is the facsimile images, a Sequence of Bit Strings, each encoding a single page of Group 3 facsimile data as dictated by Recommendation T.4, but filled out to a multiple of 8 bits with additional zero bits, and with each group of 8 bits reversed such that, for each page of the T.4 data:

- the 1st bit in the T.4 data becomes the 8th bit in the G3FacsimileData bit string;
- the 8th bit in the T.4 data becomes the 1st bit in the G3FacsimileData bit string;
- the 9th bit in the T.4 data becomes the 16th bit in the G3FacsimileData bit string;
- the 16th bit in the T.4 data becomes the 9th bit in the G3FacsimileData bit string, etc.

The Return-To-Control signal (defined in Recommendation T.4) shall be present at the end of each page of T.4 data.

NOTE 2 – The Number-of-pages component identifies the number of elements in the Sequence that constitutes the Data component and is thus redundant.

NOTE 3 – If the Body comprises a single such body part, its NBPs may (but need not) be conveyed by means of the envelope of the message that contains the IPM.

NOTE 4 – Where the body part has been received from a facsimile terminal, the Return-To-Control signal may be encoded in the form that it was received from the terminal. Where received over an unreliable network, the Return-To-Control signal may be subject to error.

NOTE 5 – All of the pages in any one G3 Facsimile body part are required to have the same values for the non-basic parameters. When a message is created containing facsimile data where pages have different values for these parameters (e.g. different paper size) it will be necessary to use a separate body part for each group of adjacent pages that have the same parameter values. Except where the originator of the message deliberately requests separate body parts, all consecutive pages that have the same parameter values should be placed in a single body part. In particular, when receiving pages from a facsimile terminal, if a new set of parameters are signalled, they should be compared with the parameters of the previous page; a new body part should be created only if the parameters have changed.

NOTE 6 – The G3 Facsimile body part is primarily intended for the conveyance of bitmap graphical data. Recommendations T.4 and T.30 provide encodings which could, in principle, allow this body part to be used for a variety of other applications (e.g. file-transfer). However, more effective interworking will be achieved if these data types are encoded using more specific body parts (such as the file-transfer body part, see 7.4.12), or content-types (such as EDI-messaging).

7.4.3 G4 Class 1

A **G4 Class 1** body part represents a final-form document of the sort that is processable by Group 4 Class 1 facsimile terminals. It comprises a Sequence of interchange data elements, defined in ITU-T Rec. T.415 | ISO/IEC 8613-5, which describe the document's layout structure.

G4Class1BodyPart ::= SEQUENCE OF Interchange-Data-Element

g4-class1-body-part EXTENDED-BODY-PART-TYPE ::= {
DATA {G4Class1BodyPart IDENTIFIED BY id-et-g4-class1} }

7.4.4 Teletex

A **Teletex** body part represents a Teletex document. It has Parameters and Data components.

TeletexBodyPart ::= SEQUENCE {

parameters TeletexParameters,
data TeletexData}

teletex-body-part EXTENDED-BODY-PART-TYPE ::= {

PARAMETERS {TeletexParameters IDENTIFIED BY id-et-teletex},
DATA {TeletexData IDENTIFIED BY id-et-teletex} }

TeletexParameters ::= SET {

number-of-pages [0] INTEGER OPTIONAL,
telex-compatible [1] BOOLEAN DEFAULT FALSE,
non-basic-parameters [2] TeletexNonBasicParameters OPTIONAL}

TeletexData ::= SEQUENCE OF TeletexString

The Parameters component comprises the following parameters:

- a) **Number-of-pages (O):** Identifies the number of pages of Teletex text present in the Data component. A non-negative Integer.
- b) **Telex-compatible (D *false*):** Indicates whether the document in the Data component is telex-compatible. A Boolean.

If this parameter has the value *true*, every Teletex String in the Data component shall be restricted to the ITA2 character set. No line shall exceed 69 characters in length.

- c) **Non-basic-parameters (C)**: Identifies the NBPs for Teletex that characterize the Data component. A Teletex NBPs descriptor.

This conditional parameter may be absent if the data component is of basic Teletex type. If the data component is of a non-basic type, it shall be present if (but not only if) the Body contains two or more Teletex body parts. The absence of this parameter when the data component is of a non-basic type is discouraged.

NOTE 1 – Its absence in these conditions provides compatibility with Recommendation X.420 (1984).

The Data component is the document, a Sequence of Teletex Strings, each of which encodes one of its pages. The text of every page (including the first page of the document) shall be introduced either by Form Feed and Carriage Return or by Carriage Return and Form Feed. This sequence shall be preceded by Identify Graphic Subrepertoire under the conditions stated in Recommendation T.61.

NOTE 2 – The Number-of-pages component identifies the number of elements in the Sequence that constitutes the Data component, and is thus redundant.

NOTE 3 – If the Body comprises a single such body part, its NBPs may (but need not) be conveyed by means of the envelope of the message that contains the IPM.

NOTE 4 – The initial Form Feed and Carriage Return on each page of a Teletex document is required by Recommendation T.61.

7.4.5 Videotex

A **Videotex** body part represents Videotex data. It has Parameters and Data components.

```
VideotexBodyPart ::= SEQUENCE {
    parameters    VideotexParameters,
    data          VideotexData}

videotex-body-part EXTENDED-BODY-PART-TYPE ::= {
    PARAMETERS {VideotexParameters IDENTIFIED BY id-ep-videotex},
    DATA      {VideotexData IDENTIFIED BY id-et-videotex} }

VideotexParameters ::= SET {
    syntax [0] VideotexSyntax OPTIONAL}

VideotexData ::= VideotexString
```

The Parameters component comprises the following parameters:

- **Syntax (O)**: Identifies the syntax of the Data component. In the parameter's absence, the syntax shall be considered unspecified.

```
VideotexSyntax ::= INTEGER {
    ids          (0),
    data-syntax1 (1),
    data-syntax2 (2),
    data-syntax3 (3)}
```

This parameter may assume any one of the following values, each of which denotes as follows one of the Videotex syntaxes defined in Recommendations T.100 and T.101:

- i) *ids*: The IDS syntax.
- ii) *data-syntax1*: Data Syntax 1.
- iii) *data-syntax2*: Data Syntax 2.
- iv) *data-syntax3*: Data Syntax 3.

The Data component is the Videotex data, a Videotex String. It shall conform to the Videotex syntax denoted by the Syntax parameter.

7.4.6 Encrypted

An **Encrypted** body part represents the result of encrypting a body part of a type defined by this Specification. It has Parameters and Data components.

```
EncryptedBodyPart ::= SEQUENCE {
    parameters    EncryptedParameters,
    data          EncryptedData}

encrypted-body-part EXTENDED-BODY-PART-TYPE ::= {
    PARAMETERS {EncryptedParameters IDENTIFIED BY id-ep-encrypted},
    DATA      {EncryptedData IDENTIFIED BY id-et-encrypted} }
```

EncryptedParameters ::= SET OF ENCRYPTED-PARAMETERS.&Type -- for future standardization

ENCRYPTED-PARAMETERS ::= CLASS {&Type}

EncryptedData ::= BIT STRING -- for future standardization

The parameters of such a body part, and the encryption technique that those parameters might identify and parameterize, may be the subject of future standardization.

The Data component is the encrypted body part, a Bit String. The bits of the Bit String shall encrypt a data value of (ASN.1) type BodyPart encoded in accordance with the Basic Encoding Rules of ITU-T Rec. X.690 | ISO/IEC 8825-1.

7.4.7 Message

A **Message** body part represents an IPM and, optionally, its delivery envelope. It has Parameters and Data components.

```

MessageBodyPart ::= SEQUENCE {
    parameters    MessageParameters,
    data          MessageData}

message-body-part EXTENDED-BODY-PART-TYPE ::= {
    PARAMETERS {MessageParameters IDENTIFIED BY id-ep-message},
    DATA      {MessageData IDENTIFIED BY id-et-message} }

MessageParameters ::= SET {
    delivery-time    [0] MessageDeliveryTime OPTIONAL,
    delivery-envelope [1] OtherMessageDeliveryFields OPTIONAL}

MessageData ::= IPM
  
```

The Parameters component comprises the following parameters:

- a) **Delivery-time** (O): The date and time the IPM was delivered. The presence of this component in the absence of the Delivery-envelope component is discouraged.
- b) **Delivery-envelope** (O): The IPM's other message delivery fields. The presence of this component in the absence of the Delivery-time component is discouraged.

The Data component is the IPM.

Including one IPM in another, as described in the present clause, is called **forwarding** that IPM. The enclosing IPM is called the **forwarding IPM**, the enclosed IPM the **forwarded IPM**.

NOTES

- 1 The possible future inclusion of a message identifier in the Parameters component may be the subject of future standardization. Its present omission provides compatibility with Recommendation X.420 (1984).
- 2 That the IPM and purported delivery envelope of a Message body part are, in any sense, genuine is unverified.

7.4.8 Mixed-mode

A **Mixed-mode** body part represents a final-form document of the sort that is processable by mixed-mode Teletex terminals and Group 4 Classes 2 and 3 facsimile terminals. It comprises a Sequence of interchange data elements, defined in ITU-T Rec. T.415 | ISO/IEC 8613-5, which describe the document's layout structure.

```

MixedModeBodyPart ::= SEQUENCE OF Interchange-Data-Element

mixed-mode-body-part EXTENDED-BODY-PART-TYPE ::= {
    DATA {MixedModeBodyPart IDENTIFIED BY id-et-mixed-mode} }
  
```

7.4.9 Bilaterally Defined

A **Bilaterally Defined** body part represents an information object whose semantics and abstract syntax are *bilaterally agreed* by the IPM's originator and all of its potential recipients. It comprises an Octet String.

```

BilaterallyDefinedBodyPart ::= OCTET STRING

bilaterally-defined-body-part EXTENDED-BODY-PART-TYPE ::= {
    DATA {BilaterallyDefinedBodyPart IDENTIFIED BY id-et-bilaterally-defined} }
  
```

NOTE – The use of this body part type is discouraged. It predates the Extended body part type and is retained for backward compatibility with Recommendation X.420 (1984). The Extended body part type provides the same capabilities and more, and its use is preferred, e.g. because such use clearly distinguishes between the body parts defined by one community of users and those defined by another.

7.4.10 Nationally Defined

A **Nationally Defined** body part represents an information object whose semantics and abstract syntax are nationally defined by a country whose identity is *bilaterally agreed* by the IPM's originator and all of its potential recipients. It comprises an Any.

NATIONAL-BODY-PARTS ::= CLASS {&Type}

NationallyDefinedBodyPart ::= NATIONAL-BODY-PARTS.&Type
-- Provided for Historic reasons. Use is strongly deprecated.

nationally-defined-body-part EXTENDED-BODY-PART-TYPE ::= {
DATA {NationallyDefinedBodyPart IDENTIFIED BY id-et-nationally-defined} }

NOTES

1 This body part type is intended for use in domestic communication where the country in question is implicitly that of the originator and all of the potential recipients.

2 The use of this body part type is discouraged. It predates the Extended body part type and is retained for backward compatibility with Recommendation X.420 (1984). The Extended body part type provides the same capabilities and more, and its use is preferred, e.g. because such use clearly distinguishes between the body parts defined by one country and those defined by another.

7.4.11 General Text

A **General Text** extended body part represents character text of a general nature. It has Parameters and Data components.

general-text-body-part EXTENDED-BODY-PART-TYPE ::= {
PARAMETERS {GeneralTextParameters IDENTIFIED BY id-ep-general-text},
DATA {GeneralTextData IDENTIFIED BY id-et-general-text} }

GeneralTextParameters ::= SET OF CharacterSetRegistration

GeneralTextData ::= GeneralString

The Parameters component comprises a list of the character set registrations that are or may be present in the Data component. Each character set is represented by the registration number defined in the registration of that character set, registered in accordance with ISO 2375.

The implicit character sets (registration numbers 2 and 1) specified by the ASN.1 Basic Encoding Rules should be present in the Parameters component if they are used.

CharacterSetRegistration ::= INTEGER (1..32767)

The Data component comprises a single General String. Character set designators other than those for character sets defined in the Parameters component shall not be used.

Each General String shall be encoded using 8-bit encoding (not 7-bit).

Within the Data component, lines may be of any length. Whenever the component is rendered (e.g. displayed to or printed for a user), all (rather than only a part) of the text must be communicated (e.g. lines may be folded but shall not be truncated).

For this extended body part type, extended EITs are defined [for the purposes of item c) of 20.4], as follows. One EIT is used for each character set the Parameters component has explicitly identified. It is denoted by the Object Identifier assigned to that character set.

This specification acts as the registration authority for such Object Identifiers, as follows. All the Object Identifiers are allocated as leaves immediately under the single vertex representing this registration authority (id-cs-eit-authority). The Object Identifier component identifying the character set represented by the leaf is the registration number of that character set as allocated in accordance with ISO 2375.

Example – The extended EITs for Latin Alphabet No. 1 (ISO 8859-1) are {id-cs-eit-authority 1} for the C0 set, {id-cs-eit-authority 6} for the G0 set and {id-cs-eit-authority 100} for the G1 set.

NOTES

1 It is preferred that the list of characters sets within the Parameters component includes only the registration numbers of those character sets which are actually used within the Data component.

2 The ASN.1 Basic Encoding Rules (ITU-T Rec. X.690-Series | ISO/IEC 8825) provide default character sets designated and invoked for G0 and C0. These rules require that any other character set designators are inserted within the encoding of the General String. However, since different editions of these rules specify different defaults, it is recommended that all required G sets (even sets 2 or 6 for IA5) are explicitly designated. The G sets then have to be invoked using locking shift or single shift control functions.

3 The registration numbers and the associated escape sequences for the character set designators are defined in the *ISO International Register of Coded Character Sets To Be Used With Escape Sequences*. This is the register established in accordance with ISO 2375.

7.4.12 File Transfer

A **File Transfer** body part represents an information object used to convey the contents, and optionally the attributes, of a stored file. The file transfer body part is based on the file model defined in ISO 8571-2. It has Parameters and Data components.

```
file-transfer-body-part EXTENDED-BODY-PART-TYPE ::= {
  PARAMETERS (FileTransferParameters IDENTIFIED BY id-ep-file-transfer),
  DATA      (FileTransferData IDENTIFIED BY id-et-file-transfer) }

FileTransferParameters ::= SEQUENCE {
  related-stored-file [0] RelatedStoredFile OPTIONAL,
  contents-type       [1] ContentTypeParameter DEFAULT document-type:
    { document-type-name {iso standard 8571 document-type (5)
      unstructured-binary (3)} },
  environment         [2] EnvironmentParameter OPTIONAL,
  compression         [3] CompressionParameter OPTIONAL,
  file-attributes      [4] FileAttributes OPTIONAL,
  extensions           [5] ExtensionsField OPTIONAL }
```

FileTransferData ::= SEQUENCE OF EXTERNAL

- This conveys a sequence of data values representing file contents;
- The rules for generating this sequence are implied by the value of the contents-type parameter.
- If the data values are a sequence of values of ASN.1 type Octet String, or a sequence of octet-aligned values
- not defined using ASN.1, then the encoding of each data value in the External may occupy either the
- single-ASN1-type alternative (as an explicitly tagged Octet String) or the octet-aligned alternative (as
- an implicitly tagged Octet String). Otherwise, the encoding should occupy the single-ASN1-type alternative.

NOTE – A number of elements of the FileTransferParameters are of type GraphicString. The ASN.1 Basic Encoding Rules specify that the encoding for these strings may include ISO/IEC 2022 escape sequences to designate and invoke graphic character sets. A default G0 set is provided, but any other G sets that are required must be explicitly designated and invoked. This requires that implementations accept on reception various combinations of escape sequences and/or shift functions even if only IA5 characters are supported. On origination, it is recommended that the default is ignored and that all required sets are explicitly designated and invoked using the appropriate escape sequences and locking shift functions. For implementations wishing to originate IA5 characters, it is recommended that character set 6 be invoked and designated (the octets required to achieve this are ESC, 2/8, 4/2, LS0).

7.4.12.1 Related Stored File parameter

The Related Stored File parameter indicates to the recipient any intended relationship between the file in this body part and any file(s) held by the recipient. Stored files may be identified either by pathname or by reference to previous MHS messages sent. Explicit relationships with stored files that may be indicated include:

- unspecified;
- a new file may be created using the contents in this body part;
- the contents of an existing file may be replaced by the contents in this body part;
- an existing file may be extended using the contents of this body part.

The syntax for this parameter is:

```
RelatedStoredFile ::= SET OF SEQUENCE {
  file-identifier FileIdentifier,
  relationship    Relationship DEFAULT explicit-relationship: unspecified }

FileIdentifier ::= CHOICE {
  pathname-and-version [0] PathnameandVersion,
  cross-reference       [1] CrossReference }
```

COPYRIGHT

```

PathnameandVersion ::= SEQUENCE {
    pathname    [0] Pathname-Attribute,
    file-version [1] GraphicString OPTIONAL}

CrossReference ::= SEQUENCE {
    application-cross-reference [0] OCTET STRING,
    message-reference           [1] MessageReference OPTIONAL,
    body-part-reference         [2] INTEGER OPTIONAL}

MessageReference ::= SET {
    user [0] ORName OPTIONAL,
    -- Defined in 8.5.5 of ITU-T Rec. X.411 | ISO/IEC 10021-4
    user-relative-identifier [1] PrintableString}

Relationship ::= CHOICE {
    explicit-relationship [0] ExplicitRelationship,
    descriptive-relationship [1] GraphicString}

ExplicitRelationship ::= INTEGER {
    unspecified (0),
    new-file (1),
    replacement (2),
    extension (3)}

```

The pathname option is intended for use in a manner consistent with ISO 8571-2, as amended by Amendment 1. It is a sequence of elements, each of which represents a name component. When more than one element is encoded, the first element shall be the file name and the remaining elements shall be concatenated to represent the file name prefix.

NOTE 1 – ISO 8571-2/Amd.1 renamed the “filename” attribute in ISO 8571-2 to the “pathname” attribute.

A message reference has the following components:

- a) **User (C)**: Identifies the user who originated the referenced message. One of the user’s OR-names. This conditional component shall be present unless the reference is an IPM identifier which does not contain a User component.
- b) **User-relative-identifier (M)**: Unambiguously identifies a message, distinguishing it from all other messages that the user who is identified by the User component originates. A Printable String of from zero to a prescribed number of characters. A length of zero is discouraged.

NOTE 2 – The MessageReference shares the same value set with the IPMIdentifier, EDIMIdentifier and VMIdentifier. Hence a file transfer body part is capable of referencing IPM, EDIM or VM messages.

A body part reference uniquely identifies a body part within a message. It is for use when referencing a message with a content type which includes body part references.

7.4.12.2 Contents Type parameter

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

```

ContentsTypeParameter ::= Contents-Type-Attribute

Contents-Type-Attribute ::= CHOICE {
    document-type [0] SEQUENCE {
        document-type-name Document-Type-Name,
        parameter [0] DOCUMENT-PARAMETER.&Type OPTIONAL},
    -- The actual types to be used for values of the parameter field
    -- are defined in the named document type.
    constraint-set-and-abstract-syntax [1] SEQUENCE {
        constraint-set-name Constraint-Set-Name,
        abstract-syntax-name Abstract-Syntax-Name}}

Document-Type-Name ::= OBJECT IDENTIFIER

DOCUMENT-PARAMETER ::= CLASS {&Type}

Constraint-Set-Name ::= OBJECT IDENTIFIER

Abstract-Syntax-Name ::= OBJECT IDENTIFIER

```

The value is either a document-type name (optionally with parameters of type ANY) or a pair of abstract syntax name and constraint set name. Each of these names is an Object Identifier.

The concepts of document-type and constraint set are described fully in ISO 8571-1 and ISO 8571-2. Examples of document types which may be used in this body part are:

- a) unstructured text file (FTAM-1);
- b) unstructured binary file (FTAM-3);
- c) sequential binary file (FTAM-4).

7.4.12.3 Environment parameter

The Environment parameter describes the environment (e.g. machine, operating system, and application) from which the file originated. It has the following syntax:

```
EnvironmentParameter ::= SEQUENCE {
    application-reference [0] GeneralIdentifier OPTIONAL,
    machine                [1] GeneralIdentifier OPTIONAL,
    operating-system       [2] OBJECT IDENTIFIER OPTIONAL,
    user-visible-string    [3] SEQUENCE OF GraphicString OPTIONAL }

GeneralIdentifier ::= CHOICE {
    registered-identifier [0] OBJECT IDENTIFIER,
    descriptive-identifier [1] SEQUENCE OF GraphicString }
```

The application-reference field is intended to be used for identifying application programs and versions. The machine field is intended to be used for executable code modules to indicate hardware platforms. The operating-system field is intended to be used to identify the operating system of the processor from which the file originated.

7.4.12.4 Compression parameter

The Compression parameter describes the compression type if the file is transferred in a compressed mode.

```
CompressionParameter ::= SEQUENCE {
    compression-algorithm-id [0]
        COMPRESSION-ALGORITHM.&id ({CompressionAlgorithmTable}),
    compression-algorithm-param [1]
        COMPRESSION-ALGORITHM.&Type ({CompressionAlgorithmTable} {@compression-algorithm-id})}

COMPRESSION-ALGORITHM ::= TYPE-IDENTIFIER

CompressionAlgorithmTable COMPRESSION-ALGORITHM ::= { ... }
```

7.4.12.5 File Attributes parameter

The File Attributes parameter conveys values of any of a set of optional file attributes. When the recipient is to create a new file, these values are to be used in establishing the initial file attributes.

NOTE – Transfer of an attribute value to a recipient should be interpreted as a request only; no particular recipient behaviour is guaranteed as a result.

The file attributes are technically aligned with ISO 8571-2. The semantic descriptions of these attributes in ISO 8571-2 take precedence over the abbreviated descriptions given below. The file attributes which can be conveyed in this parameter are:

- pathname;
- permitted actions;
- storage account;
- date and time of creation;
- date and time of last modification;
- date and time of last read access;
- date and time of last attribute modification;
- identity of creator;
- identity of last modifier;
- identity of last reader;

- identity of last attribute modifier;
- availability;
- object size;
- future object size;
- access control;
- legal qualifications;
- private use;
- attribute-extensions.

The syntax for the file attributes parameter is as follows:

FileAttributes ::= SEQUENCE {	
pathname	Pathname-Attribute OPTIONAL,
permitted-actions	[1] Permitted-Actions-Attribute OPTIONAL,
storage-account	[3] Account-Attribute OPTIONAL,
date-and-time-of-creation	[4] Date-and-Time-Attribute OPTIONAL,
date-and-time-of-last-modification	[5] Date-and-Time-Attribute OPTIONAL,
date-and-time-of-last-read-access	[6] Date-and-Time-Attribute OPTIONAL,
date-and-time-of-last-attribute-modification	[7] Date-and-Time-Attribute OPTIONAL,
identity-of-creator	[8] User-Identity-Attribute OPTIONAL,
identity-of-last-modifier	[9] User-Identity-Attribute OPTIONAL,
identity-of-last-reader	[10] User-Identity-Attribute OPTIONAL,
identity-of-last-attribute-modifier	[11] User-Identity-Attribute OPTIONAL,
object-availability	[12] Object-Availability-Attribute OPTIONAL,
object-size	[13] Object-Size-Attribute OPTIONAL,
future-object-size	[14] Object-Size-Attribute OPTIONAL,
access-control	[15] Access-Control-Attribute OPTIONAL,
legal-qualifications	[16] Legal-Qualification-Attribute OPTIONAL,
private-use	[17] Private-Use-Attribute OPTIONAL,
attribute-extensions	[22] Attribute-Extensions OPTIONAL }

The types of all the above components are defined below or imported from ISO 8571-4.

7.4.12.5.1 Pathname attribute

The pathname attribute provides a file name.

```
Pathname-Attribute ::= CHOICE {
    incomplete-pathname [0] Pathname,
    complete-pathname   [23] Pathname }
```

7.4.12.5.2 Permitted actions attribute

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

7.4.12.5.3 Storage account attribute

The storage account attribute identifies the accountable authority responsible for accumulated file storage charges.

```
Account-Attribute ::= CHOICE {
    no-value-available [0] NULL,
    -- Indicates partial support of this attribute
    actual-values       Account }
```

```
Account ::= GraphicString
```

7.4.12.5.4 Date and time attributes

The date and time of creation attribute indicates when the file was created.

The date and time of last modification attribute indicates when the contents of the file were last modified.

The date and time of last read access attribute indicates when the contents of the file were last read.

The date and time of last attribute modification attribute indicates when the attributes of the file were last modified.

7.4.12.5.5 Identity attributes

The identity of creator, identity of last modifier, identity of last reader, and identity of last attribute modifier attributes identify the user(s) who created, last modified, and last read the file.

```
User-Identity-Attribute ::= CHOICE {
    no-value-available [0] NULL,
    -- Indicates partial support of this attribute.
    actual-values       User-Identity }
```

```
User-Identity ::= GraphicString
```

7.4.12.5.6 Availability attribute

The availability attribute indicates whether the file had (or should have) immediate or deferred availability (e.g. whether it was stored on permanently mounted or demountable storage media).

7.4.12.5.7 Object size attributes

The object size attribute is set to the nominal size in octets of the complete file.

The future object size attribute indicates the nominal size in octets to which the file may grow as a result of modification and extension.

7.4.12.5.8 Access control attribute

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

```
Access-Control-Attribute ::= CHOICE {
    no-value-available [0] NULL,
    -- Indicates partial support of this attribute.
    actual-values       [1] SET OF Access-Control-Element }
```

-- The semantics of this attribute are described in ISO 8571-2

```
Access-Control-Element ::= SEQUENCE {
    action-list           [0] Access-Request,
    concurrency-access    [1] Concurrency-Access OPTIONAL,
    identity              [2] User-Identity OPTIONAL,
    passwords             [3] Access-Passwords OPTIONAL,
    location              [4] Application-Entity-Title OPTIONAL }
```

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

```
Access-Passwords ::= SEQUENCE {
    read-password       [0] Password,
    insert-password     [1] Password,
    replace-password    [2] Password,
    extend-password     [3] Password,
    erase-password      [4] Password,
    read-attribute-password [5] Password,
    change-attribute-password [6] Password,
    delete-password     [7] Password,
    pass-passwords      [8] Pass-Passwords,
    link-password       [9] Password }
```

```
Password ::= CHOICE { graphic-string GraphicString, octet-string OCTET STRING }
```

```
Pass-Passwords ::= SEQUENCE OF Password
```

```
Application-Entity-Title ::= SEQUENCE {
    ap-title    AP-title,
    ae-qualifier AE-qualifier }
```

-- AP-title and AE-qualifier are defined in ITU-T Rec. X.227 | ISO/IEC 8650-1

7.4.12.5.9 Legal qualifications attribute

The legal qualifications attribute conveys information about the legal status of the file and its use.

7.4.12.5.10 Private use attribute

The meaning of the private use attribute is not defined.

7.4.12.5.11 Attribute extensions

The attribute extensions attribute allows for the inclusion of additional attributes in a manner consistent with Amendment 1 of ISO 8571-2 and ISO 8571-4.

7.4.12.6 Extensions parameter

The extensions parameter conveys information accommodated by no other parameter of the file transfer body part. The syntax and usage of this field are the same as those of the extensions heading field specified in 7.2.17.

7.4.12.7 File Transfer Body Part data

The data component of the body part contains the file contents being transferred.

The syntax for representation of these contents is implied by the contents-type parameter. When this parameter specifies a document type, the corresponding document type definition describes how to construct a sequence of Externals to convey the data values representing the file contents, and identifies the necessary abstract syntax(es). The encoding of each data value in the External is described in 7.4.12. When the contents-type parameter specifies a constraint set and abstract syntax, the contents comprise a sequence of one or more data values from the identified abstract syntax.

The encoding shall be based on the transfer syntax specified as part of the document type definition, if any, or on the ASN.1 basic encoding rules otherwise.

7.4.12.8 Encoded Information type

For this extended body part type, an extended EIT is defined [for the purposes of item c) of 20.4] by the object identifier id-eit-file-transfer. This value shall be used in all instances of a message containing the File Transfer body part.

Additional EITs may optionally be derived from each of the parameter components Contents Type Parameter, Environment Parameter and Compression Parameter. If used, these shall be derived as follows:

- a) If the Contents Type Parameter is encoded as document-type or specified by default, the additional EIT is the Object Identifier assigned to that document type. If the Contents Type Parameter is encoded as constraint-set-and-abstract-syntax, two additional EITs are defined, having the values of the Object Identifiers assigned to the constraint-set-name and abstract-syntax-name.
- b) If the Environment Parameter contains an application-reference which is encoded as an Object Identifier, the additional EIT is the Object Identifier assigned to that application.
- c) If the Compression Parameter is present, the additional EIT is the Object Identifier assigned to the compression-algorithm-id.

NOTES

1 When a file-transfer body part is being submitted by a UA which did not originally encode it (e.g. when forwarding a message), the information necessary to encode the additional EITs may not be available. In this case, only the primary EIT id-eit-file-transfer needs to be used.

2 The use of additional EITs where the recipient has not registered to allow reception of those EITs will cause non-delivery. Practical use of additional EITs will therefore be subject to bilateral agreement between originator and recipient.

7.4.13 Voice

A **Voice** body part represents speech, or other types of audio data such as music. It has Parameters and Data components.

NOTE 1 – The voice body part defined here replaces the basic voice body part specified in CCITT Rec. X.420 (1992) | ISO/IEC 10021-7:1990.

Each instance of this body part in an IPM carries a single voice encoded message. The *voice-encoding-type* is specified in the voice parameters component.

```
voice-body-part EXTENDED-BODY-PART-TYPE ::= {
  PARAMETERS    {VoiceParameters IDENTIFIED BY id-ep-voice},
  DATA          {VoiceData IDENTIFIED BY id-et-voice} }
```

```

VoiceParameters ::= SEQUENCE {
    voice-message-duration      [0] INTEGER OPTIONAL, -- In seconds
    voice-encoding-type         [1] OBJECT IDENTIFIER,
    supplementary-information    [2] IA5String OPTIONAL }

```

VoiceData ::= OCTET STRING

The Parameters component contains the following components:

- a) **Voice-message-duration** (O): The duration of the voice encoded data component in seconds.
- b) **Voice-encoding-type** (M): An object identifier which identifies the voice encoding applied to the data portion of this body part.
- c) **Supplementary-information** (O): Conveys additional information to advise the recipient of the content of the voice encoded data component.

Four values for voice-encoding-type are defined by this Specification to identify voice encodings commonly used with personal computers:

- a) *id-voice-11khz-sample*: Identifies encoding as 8-bit linear-law samples at a nominal 11 Khz (11 025 Hz) sampling frequency. Each sample comprises a signed integer value in the range -128 to +127, bearing a linear relationship to the amplitude of the analogue wave-form. Each sample is encoded as a single octet, containing a 2's complement representation of the integer value, with the least significant bit of the value in the least significant bit of the octet.
- b) *id-voice-22khz-sample*: Identifies encoding as 8-bit linear-law samples at a nominal 22 Khz (22 050 Hz) sampling frequency. Each sample comprises a signed integer value in the range -128 to +127, bearing a linear relationship to the amplitude of the analogue wave-form. Each sample is encoded as a single octet, containing a 2's complement representation of the integer value, with the least significant bit of the value in the least significant bit of the octet.
- c) *id-voice-cd-quality*: Identifies encoding of 'Compact Disc' quality, as specified in IEC 908. Sampling occurs at a rate of 44 100 Hz. Each sample comprises two 16-bit values, for left and right channels of a stereo pair, and is encoded as four octets in the following order: 8 least significant bits of 'left' channel; 8 most significant bits of 'left' channel; 8 least significant bits of 'right' channel; 8 most significant bits of 'right' channel. Within each octet, the most significant bit of the sample is placed in the most significant bit of the octet, and the least significant bit of the sample is placed in the least significant bit of the octet. The length of the data shall be a multiple of 4 octets.
- d) *id-voice-g711-mu-law*: Identifies Pulse Code Modulation (PCM) encoding at 8000 samples/s as specified in Recommendation G.711, using μ -law. Each value specified by G.711 comprises 8 bits; each value is encoded as a single octet, with bit No. 1 (polarity bit) of the sample in the most significant bit of the octet, and bit No. 8 (the least significant bit) encoded in the least significant bit of the octet.

NOTE 2 – The ability of individual implementations to record or replay voice body parts will be limited by hardware capabilities. In some cases the sampling rate will be slightly higher or lower than the nominal rate specified, or interpolation will be required to match 11 Khz samples with hardware capable of only 22 Khz.

Two further values for voice-encoding-type are defined to identify voice encodings commonly used in digital telephony:

- a) *id-voice-g726-32k-adpcm*: Identifies 32 kbit/s Adaptive Differential Pulse Code Modulation (ADPCM) encoding as specified in Recommendation G.726. The 4-bit code words of the G.726 encoding shall be packed into the octets of the OCTET STRING as follows: the first code word is placed in the four least significant bits of the first octet, with the least significant bit of the code word in the least significant bit of the octet; the second code word is placed in the four most significant bits of the first octet, with the most significant bit of the code word in the most significant bit of the octet. Subsequent pairs of code words shall be packed in the same way into successive octets, with the first code word of each pair placed in the least significant four bits of the octet. It is preferred that the voice sample be extended with silence such that the encoded value comprises an even number of code words. However, if the voice sample comprises an odd number of code words, then the last code word shall be discarded.
- b) *id-voice-g728-16k-ld-celp*: Identifies 16 kbit/s Low-Delay Code Excited Linear Prediction (LD-CELP) encoding as specified in Recommendation G.728. The 10-bit code words of the G.728 encoding shall be packed into the octets of the OCTET STRING by considering groups of four code words as follows.

The eight least significant bits of the first code word are placed in the first octet, with the least significant bit of the code word as the least significant bit of the octet. The two most significant bits of the first code word are placed in the two least significant bits of the second octet, with the less significant of the bits from the code word as the least significant bit of the octet. The six least significant bits of the second code word are placed in the six most significant bits of the second octet, with the most significant bit of those taken from the code word as the most significant bit of the octet. The four most significant bits of the second code word are placed in the four least significant bits of the third octet, with the least significant of the bits from the code word as the least significant bit of the octet. The four least significant bits of the third code word are placed in the four most significant bits of the third octet, with the most significant bit of those taken from the code word as the most significant bit of the octet. The six most significant bits of the third code word are placed in the six least significant bits of the fourth octet, with the least significant of the bits from the code word as the least significant bit of the octet. The two least significant bits of the fourth code word are placed in the two most significant bits of the fourth octet, with the most significant bit of those taken from the code word as the most significant bit of the octet. The eight most significant bits of the fourth code word are placed in the fifth octet, with the most significant bit of the code word as the most significant bit of the octet.

Subsequent quadruplets of code words shall be packed in the same way into successive groups of five octets, with the eight least significant bits of the first code word of each group placed in the first octet. It is preferred that the voice sample be extended with silence such that the encoded value comprises a multiple of four of code words. However, if the voice sample comprises a number of code words which is not a multiple of four, then the last remaining code words (i.e. those in excess of an exact multiple of four) shall be discarded.

The Data component is the digital encoding of the voice, an Octet String. The encoding is specified by the value of the voice-encoding-type component of the Parameters.

For this extended body part type, extended EITs are defined for the purposes of item c) in 20.4. One EIT is defined for each voice body part; it comprises the value of the voice-encoding-type component of the Parameters.

7.4.14 Report

A **Report** body part represents an MTS Report. It has a Data component.

```
report-body-part EXTENDED-BODY-PART-TYPE ::= {
  DATA {ReportDeliveryArgument IDENTIFIED BY id-et-report} }
```

The Data component is the argument of the MTS Report-delivery abstract-operation (see 8.3.1.2 of ITU-T Rec. X.411 | ISO/IEC 10021-4).

If returned-content is present in the MTS Report, it is a local matter whether it is copied into the report body part.

NOTE – Forwarding of delivery reports containing returned content may present a security risk. It is desirable that UAs generating the report body part provide an option to allow the user to choose whether to include the Returned Content.

7.4.15 Notification

A **Notification** body part represents an *IPN* and, optionally, its delivery envelope. It has Parameters and Data components.

```
notification-body-part EXTENDED-BODY-PART-TYPE ::= {
  PARAMETERS {MessageParameters IDENTIFIED BY id-ep-notification},
  DATA {IPN IDENTIFIED BY id-et-notification} }
```

The Parameters component is identical to that for a message body part and is defined in 7.4.7.

The Data component is the *IPN*.

If *returned-ipm* is present in an *NRN*, it is a local matter whether it is copied into the notification body part.

NOTE – Forwarding of notifications containing returned IPMs may present a security risk. It is desirable that UAs generating the notification body part provide an option to allow the user to choose whether to include the Returned IPM.

7.4.16 Forwarded content

A **Forwarded Content** body part represents a message (not necessarily an IPM) which has been previously transferred by the MTS. It has Parameters and Data components.

A family of these body parts is defined, one for each possible content-type. It is therefore possible to forward a message of any type, using the appropriate forwarded content body part.

```
ContentBodyParts EXTENDED-BODY-PART-TYPE ::= {
  content-body-part { id-mct-p2-1984 } |
  content-body-part { id-mct-p2-1988 } |
  content-body-part { id-mct-pedi },
  -- any other body part defined by an instance of content-body-part --
  ... }
```

These body parts are defined using the following template:

```
content-body-part {ExtendedContentType:content-type} EXTENDED-BODY-PART-TYPE ::= {
  PARAMETERS {ForwardedContentParameters IDENTIFIED BY
    {id-ep-content -- concatenated with content-type -- }},
  DATA      {Content IDENTIFIED BY {id-et-content -- concatenated with content-type -- } }

ForwardedContentParameters ::= SET {
  delivery-time      [0] MessageDeliveryTime OPTIONAL,
  delivery-envelope  [1] OtherMessageDeliveryFields OPTIONAL,
  mts-identifier      [2] MessageDeliveryIdentifier OPTIONAL }
```

To realize an instance of this family of body parts, the values of Object Identifiers to identify the Parameters and Data components are constructed by concatenating the prefixes id-ep-content and id-et-content respectively with the Object Identifier defined for the content-type of the message to be forwarded. For example, to forward a message of the inner-envelope content-type (see ITU-T Rec. X.411 | ISO/IEC 10021-4), the parameters would be identified by { id-ep-content id-cont-inner-envelope } or { 2 6 1 11 17 2 6 3 3 1 }:

```
inner-envelope-content-body-part EXTENDED-BODY-PART-TYPE ::=
  content-body-part {id-cont-inner-envelope }
```

The Parameters component comprises the following parameters:

- a) **Delivery-time** (O): The date and time the original message was delivered.
- b) **MTS-identifier** (O): The MTS identifier assigned to the original message.
- c) **Delivery-envelope** (O): The original message's other message delivery fields.

The Data component is the Content of the original message.

When forwarding an IPM, the Message body part (see 7.4.7) should be used unless it is necessary to represent the message in precisely the form that was transferred by the MTS.

NOTES

1 An example of a case where it may be necessary to use the forwarded content body part to forward an IPM is where the content of the original message is encrypted or secured by content-integrity-check. Use of the forwarded content body part in these cases allows the recipient to verify the security arguments.

2 While the components of the parameters are optional (to ease implementation in simple applications), for many applications their omission would render the body part useless.

For this extended body part type, extended EITs are defined for the purposes of item c) in 20.4. One EIT is defined for each forwarded content body part; it comprises the Object Identifier which identifies the Data component.

8 Interpersonal notifications

An **Interpersonal Notification (IPN)** is a member of a secondary class of information object conveyed between users in Interpersonal Messaging.

```
IPN ::= SET {
  -- common-fields -- COMPONENTS OF CommonFields,
  choice [0] CHOICE {
    non-receipt-fields      [0] NonReceiptFields,
    receipt-fields          [1] ReceiptFields,
    other-notification-type-fields [2] OtherNotificationTypeFields }
```

An IPN may take any of the following forms:

- a) **non-receipt notification (NRN)**: An IPN that reports its originator's failure to receive, to accept, or his delay in receiving, an IPM.

```
NRN ::= IPN (WITH COMPONENTS {
    ...,
    choice (WITH COMPONENTS { non-receipt-fields PRESENT }) })
```

- b) **receipt notification (RN)**: An IPN that reports its originator's receipt, or his expected and arranged future receipt, of an IPM.

```
RN ::= IPN (WITH COMPONENTS {
    ...,
    choice (WITH COMPONENTS { receipt-fields PRESENT }) })
```

- c) **other notification (ON)**: An IPN that reports some other event concerning an IPM.

```
ON ::= IPN (WITH COMPONENTS {
    ...,
    choice (WITH COMPONENTS { other-notification-type-fields PRESENT }) })
```

There are no ONs defined in this version of the Specification. Specific uses of ON may be defined in future versions of this Specification to support extended semantics of an IPN, such as secure notifications.

The IPM to which an IPN refers is called the **subject IPM**. Only a UA to which the subject IPM is actually delivered shall originate an IPN relating to it, and it shall originate at most one such IPN which shall be conveyed to the subject IPM's originator alone.

An actual recipient shall originate an IPN only in accordance with the Notification-requests component of the *subject recipient specifier*. The **subject recipient specifier** is that recipient specifier in the subject IPM's Heading as a result of which the subject IPM is delivered to that user.

The subject recipient specifier is determined by examining the Sequences of recipient specifiers that constitute the subject IPM's Primary, Copy, and Blind Copy Recipients heading fields. The fields are examined in the order in which they are mentioned in the preceding sentence. Within each field, the specifiers are examined in the order in which they appear there. The subject recipient specifier is the first one found whose Recipient component has, as its value, an OR-descriptor whose Formal-name component is present and has, as its value, the value of either the OR-name from the first element of the redirection-history argument (if present) or else the this-recipient-name argument of the Message Delivery operation.

NOTE – In the case where the subject IPM has been delivered as a result of DL Expansion, an IPN is not generated. This is in order to prevent inadvertent disclosure of the membership of the DL (which should be the subject of the DL policy). However, IPNs may be generated in the case of Redirection or alias OR-names.

An IPN comprises a Set of information items called **notification fields** (or **fields**), each of which is of one of the following classes:

- a) **common field**: A notification field applicable to both NRNs and RNs.
- b) **non-receipt field**: A notification field applicable to NRNs alone.
- c) **receipt field**: A notification field applicable to RNs alone.
- d) **other notification type field**: A notification field applicable to ONs alone.

The structure of an IPN is depicted in Figure 2.

The fields, in each of the above classes, that may appear in an IPN are defined and described below.

8.1 Common fields

The common fields are defined and described below:

```
CommonFields ::= SET {
    subject-ipm           SubjectIPMField,
    ipn-originator        [1] IPNOriginatorField OPTIONAL,
    ipm-intended-recipient [2] IPMIntendedRecipientField OPTIONAL,
    conversion-eits       ConversionEITsField OPTIONAL,
    notification-extensions [3] NotificationExtensionsField OPTIONAL }
```

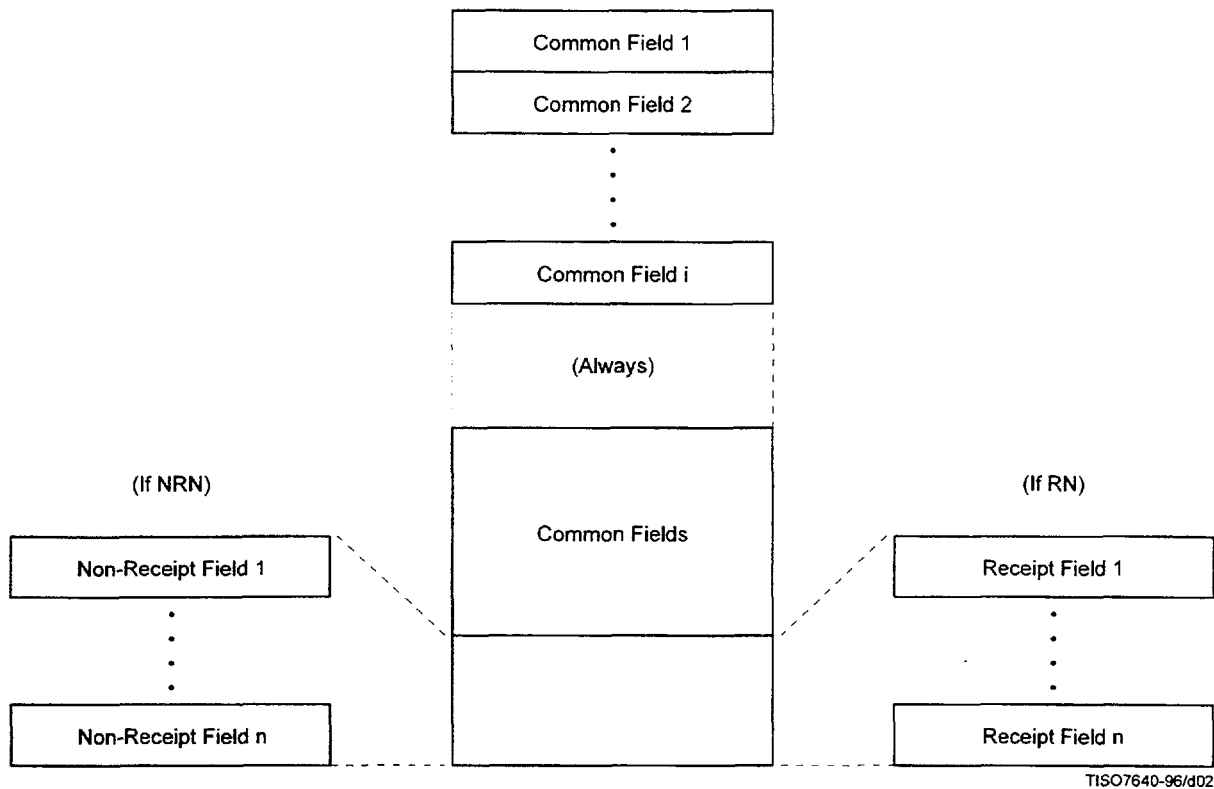


Figure 2 – An Interpersonal Notification

8.1.1 Subject IPM

The **Subject IPM** common field (M) contains the value of the This IPM field of the subject IPM. It comprises an IPM identifier.

SubjectIPMField ::= IPMIdentifier

8.1.2 IPN Originator

The **IPN Originator** common field (O) identifies the IPN's originator. It comprises an OR-descriptor.

IPNOriginatorField ::= ORDescriptor

If the IPN's originator is an intended recipient of the subject IPM, the OR-descriptor above shall be precisely that which is the value of the Recipient component of the subject recipient specifier.

8.1.3 IPM Intended Recipient

The **IPM Intended Recipient** common field (C) identifies the originally specified recipient which gave rise to the subject IPM's delivery to the IPN's originator. It comprises an OR-descriptor.

IPMIntendedRecipientField ::= ORDescriptor

The OR-descriptor above shall be precisely that which is the value of the Recipient component of the subject recipient specifier.

This conditional field shall be present if, and only if, the OR-address of the IPN's originator is different from that of the subject recipient specifier, i.e. when the IPN's originator received the message as a result of redirection, or where the subject recipient specifier contained another, non-preferred, O/R address of the same user.

8.1.4 Conversion EITs

The **Conversion EITs** common field (C) identifies the EITs of the subject IPM upon delivery to the IPN's originator. It comprises an EITs descriptor.

ConversionEITsField ::= EncodedInformationTypes

COPYRIGHT

This conditional field shall be present if, and only if, the IPM was subjected to conversion for delivery to the IPN's originator.

8.1.5 Notification Extensions

The **Notification Extensions** common field (O) allows for future extensions to the IPN.

NotificationExtensionsField ::= SET OF IPMSExtension

Notification extensions for IPMS Security are defined in Annex B. There are no other notification extensions defined in this version of this Specification.

8.2 Non-receipt fields

The non-receipt fields are defined and described below.

NonReceiptFields ::= SET {
 non-receipt-reason [0] **NonReceiptReasonField**,
 discard-reason [1] **DiscardReasonField** OPTIONAL,
 auto-forward-comment [2] **AutoForwardCommentField** OPTIONAL,
 returned-ipm [3] **ReturnedIPMField** OPTIONAL,
 nrn-extensions [4] **NRNExtensionsField** OPTIONAL;
}

8.2.1 Non-receipt Reason

The **Non-receipt Reason** non-receipt field (M) indicates why the NRN's originator has not received the subject IPM (even though it was delivered to him).

NonReceiptReasonField ::= ENUMERATED {
 ipm-discarded (0),
 ipm-auto-forwarded (1),
 ...
}

This field may assume any one of the following values:

- a) *ipm-discarded*: The IPM was discarded. This case is further illuminated by the *Discard Reason* field.
- b) *ipm-auto-forwarded*: The IPM was auto-forwarded. This case is further illuminated by the *Auto-forward Comment* field.

8.2.2 Discard Reason

The **Discard Reason** non-receipt field (C) indicates why the subject IPM was discarded (subsequent to its delivery to the NRN's originator and prior to its receipt).

[*ITU-T version:*

DiscardReasonField ::= ENUMERATED {
 ipm-expired (0),
 ipm-obsolete (1),
 user-subscription-terminated (2),
 not-used (3)
}

| *ISO/IEC version:*

DiscardReasonField ::= ENUMERATED {
 ipm-expired (0),
 ipm-obsolete (1),
 user-subscription-terminated (2),
 -- *The following value may not be supported by*
 -- *implementations of earlier versions of this Specification*
 ipm-deleted (3),
 ...
}

]

This field may assume any one of the following values:

- a) *ipm-expired*: *Auto-discard* was in effect, expired IPMs were being discarded, and the time identified by the subject IPM's Expiry Time heading field had arrived.
- b) *ipm-obsolete*: *Auto-discard* was in effect, obsolete IPMs were being discarded, and the Obsolete IPMs heading field of another IPM, delivered to the NRN's originator, identified the subject IPM.
- c) *user-subscription-terminated*: The Interpersonal Messaging subscription of the NRN's originator was terminated.

[*ITU-T version*:

- d) *not-used*: Implementations shall not generate this value. However, since this value is used by implementations conforming to ITU-T Rec. X.400-Series | ISO/IEC 10021 to indicate *ipm-deleted*, messages may be encountered that contain this value.

| *ISO/IEC version*:

- d) *ipm-deleted*: The subject IPM was deleted before receipt occurred. Where a message store is involved, deletion occurred before the IPM's status became *processed*.

]

This conditional field shall be present only if the Non-receipt Reason field has the value *ipm-discarded*. In the absence of this field the reason for discarding is not specified.

8.2.3 Auto-forward Comment

The **Auto-forward Comment** non-receipt field (C) is information pre-supplied for this purpose by the NRN's originator. It comprises a Printable String[of from zero to a prescribed number of characters (see Annex L), chosen from the Printable String character set]]. A length of zero is discouraged.

AutoForwardCommentField ::= **AutoForwardComment**

AutoForwardComment ::= PrintableString (SIZE (0..ub-auto-forward-comment))

The value of this field shall be precisely the auto-forward-comment argument of the *Change Auto-forwarding* abstract operation as a result of which the subject IPM was auto-forwarded.

This conditional field shall be present if, and only if, the Non-receipt Reason field has the value *ipm-auto-forwarded* and the auto-forward-comment argument above was supplied.

8.2.4 Returned IPM

The **Returned IPM** non-receipt field (C) is precisely the subject IPM.

ReturnedIPMField ::= **IPM**

This conditional field shall be present if, and only if, *ipm-return* is among the values of the Notification-requests component of the subject recipient specifier and the subject IPM was not subjected to conversion for delivery to the NRN's originator.

8.2.5 NRN Extensions

The **NRN Extensions** field (O) allows for future extensions to the structure of an NRN.

NRNExtensionsField ::= SET OF **IPMSExtension**

There are no NRN Extensions defined in this version of this Specification.

8.3 Receipt fields

The receipt fields are defined and described below:

ReceiptFields ::= SET {
 receipt-time {0} **ReceiptTimeField**,
 acknowledgment-mode {1} **AcknowledgmentModeField** **DEFAULT** manual,
 suppl-receipt-info {2} **SupplReceiptInfoField** **OPTIONAL**,
 rn-extensions {3} **RNExtensionsField** **OPTIONAL**}

COPYRIGHT

8.3.1 Receipt Time

The **Receipt Time** receipt field (M) identifies when the RN's originator received the subject IPM. It comprises a date and time.

ReceiptTimeField ::= Time

8.3.2 Acknowledgment Mode

The **Acknowledgment Mode** receipt field (D *manual*) identifies the manner in which the RN was originated.

AcknowledgmentModeField ::= ENUMERATED {
manual (0),
automatic (1)}

This field may assume any one of the following values:

- a) *manual*: The RN was originated by means of the *Originate RN* abstract operation.
- b) *automatic*: The RN was originated as a result of *auto-acknowledgment*.

8.3.3 Suppl Receipt Info

The **Suppl Receipt Info** receipt field (O) gives supplementary information about the receipt of the subject IPM by the RN's originator. It comprises a Printable String[of from zero to a prescribed number of characters (see Recommendation X.411), chosen from the Printable String character set[.].

SupplReceiptInfoField ::= SupplementaryInformation

8.3.4 RN Extensions

The **RN Extensions** field (O) allows for future extensions to the structure of an RN.

RNExtensionsField ::= SET OF IPMSExtension

There are no RN Extensions defined in this version of this Specification.

8.4 Other notification type fields

The other notification type fields relate to an ON. There are no ONs defined in this version of this Specification.

OtherNotificationTypeFields ::= SET OF IPMSExtension

SECTION 3 – ABSTRACT SERVICE DEFINITION

9 Overview

This section defines the abstract service that characterizes Interpersonal Messaging, and describes the environment in which that service is supplied and consumed. It does both using the abstract service definition conventions of ITU-T Rec. X.402 | ISO/IEC 10021-2.

This section covers the following topics:

- a) Primary object types;
- b) Primary port types;
- c) Abstract operations;
- d) Abstract errors;
- e) other capabilities.

10 Primary object types

The environment in which Interpersonal Messaging takes place can be modelled as an abstract object which is hereafter referred to as the **Interpersonal Messaging Environment (IPME)**.

When refined (i.e. functionally decomposed), the IPME can be seen to comprise lesser objects which interact by means of ports.

The lesser objects are referred to as the **primary objects** of Interpersonal Messaging. They include a single, central object, the *Interpersonal Messaging System (IPMS)*, and numerous peripheral objects called *Interpersonal Messaging System users (IPMS users)*.

The structure of the IPME is depicted in Figure 3.

The primary object types are defined and described below. The types of ports by means of which they interact are discussed in clause 11.

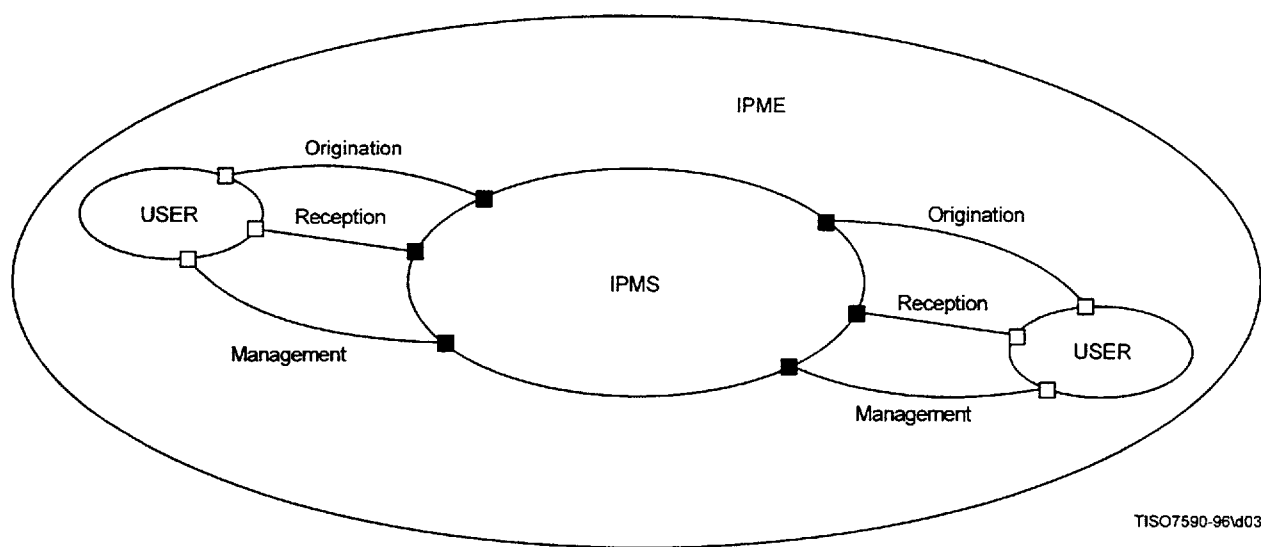


Figure 3 – The Interpersonal Messaging Environment

10.1 Interpersonal Messaging System user

An **Interpersonal Messaging System user (IPMS user)** is a user that engages in Interpersonal Messaging. An IPMS user originates, receives, or both originates and receives information objects of the types defined in Section 2.

```
ipms-user MHS-OBJECT ::= {
  INITIATES      {ipms-access-contract}
  ID             id-of-ipms-user }

ipms-access-contract CONTRACT ::= {
  INITIATOR CONSUMER OF {origination | reception | management} }
```

The IPME comprises any number of IPMS users.

NOTES

1 As its name suggests, Interpersonal Messaging is typically an activity of people. Often, therefore, this Specification uses personal pronouns (e.g. "he") to refer to IPMS users. This practice, however, is not intended to preclude other, atypical uses of Interpersonal Messaging in which IPMS users are not people.

2 For brevity, the term "user" is used throughout the rest of this Specification with the meaning of "IPMS user".

COPYRIGHT

10.2 Interpersonal Messaging System

The **Interpersonal Messaging System (IPMS)** is the object by means of which all users communicate with one another in Interpersonal Messaging.

```
ipms MHS-OBJECT ::= {
    RESPONDS      {ipms-access-contract}
    ID             id-of-ipms }
```

The IPME comprises exactly one IPMS.

11 Primary port types

The primary objects of Interpersonal Messaging are joined to and interact with one another by means of ports. These ports, which the IPMS supplies, are referred to as the **primary ports** of Interpersonal Messaging. They are of the three types defined below.

NOTE – In clause 16 to follow, the IPMS is decomposed into still lesser objects, among which is the MTS. This fact is anticipated in the present clause by the inclusion of certain MTS capabilities in the IPMS Abstract Service.

11.1 Origination

An **origination port** is the means by which a single user conveys to the IPMS messages containing information objects of the types defined in Section 2. Through such a port the user originates *interpersonal messages* and *receipt notifications*. In addition, the user may originate probes through such a port.

The IPMS supplies one origination port to each user (with the exception of indirect users served by PDAUs – see 16.5).

11.2 Reception

A **reception port** is the means by which the IPMS conveys to a single user messages containing information objects of the types defined in Section 2. Through such a port the user receives *interpersonal messages* and *interpersonal notifications*. In addition, the user may receive reports through such a port.

The IPMS supplies one reception port to each user.

11.3 Management

A **management port** is the means by which a single user changes information about himself on file with the IPMS. By means of such a port the user enables and disables *auto-discard*, *-acknowledgment*, and *-forwarding*.

The IPMS supplies one management port to each user (with the exception of indirect users served by PDAUs – see 16.5).

12 Abstract operations

The **IPMS Abstract Service** is the set of capabilities that the IPMS provides to each user by means of one origination, one reception, and one management port. Those capabilities are modelled as abstract operations, which may encounter abstract errors when invoked.

The abstract operations available at origination, reception, and management ports, respectively, are defined and described below. The abstract errors they may provoke are the subject of clause 13.

NOTES

- 1 The IPMS Abstract Service involves neither abstract bind nor abstract unbind operations.
- 2 The IPMS authenticates (i.e. establishes the identity of) the typical user before offering the IPMS Abstract Service to him. By this means it can verify, e.g. that the user is an IPMS subscriber. Authentication, where required, is implicit (rather than explicit) in the definition of the IPMS Abstract Service.
- 3 The purpose of the IPMS Abstract Service definition is not to prescribe the user interfaces of implementations of portions of the IPMS, but rather to clarify the meaning and intended use of the information objects of Section 2. A user interface need not provide commands in one-to-one correspondence to the service's abstract operations, nor indeed even divide the labour between the user and the IPMS as the service does. Also, the IPMS Abstract Service definition does not model the facilities provided by a Message Store.
- 4 In clause 16 to follow, the IPMS is decomposed into objects among which is the MTS. The present clause reflects this fact by its inclusion of various MTS-defined information items in the IPMS Abstract Service.

12.1 Origination abstract operations

The abstract operations available at an origination port are invoked by the user and performed by the IPMS.

```
origination PORT ::= {
  CONSUMER INVOKES { originate-probe | originate-ipm | originate-rn | originate-on }
  ID                id-pt-origination }
```

12.1.1 Originate Probe

The **Originate Probe** abstract operation originates a probe concerning (a class of) messages whose contents are IPMs.

```
originate-probe ABSTRACT-OPERATION ::= {
  ARGUMENT SET {
    envelope [0] ProbeSubmissionEnvelope,
    content  [1] IPM}
  RESULT SET {
    submission-identifier [0] ProbeSubmissionIdentifier,
    submission-time       [1] ProbeSubmissionTime}
  ERRORS {
    subscription-error |
    recipient-improperly-specified} }
```

This abstract operation has the following arguments:

- a) **Envelope (M)**: A probe submission envelope, whose make-up the MTS Abstract Service defines. The UA supplies all but the following envelope components, which the user provides:
 - i) the desired per-message options (i.e. per-message indicators and extensions);
 - ii) the OR-names of the intended recipients and the per-recipient options (i.e. originator report request, explicit conversion, and extensions) desired for each.
- b) **Content (M)**: An instance of the class of IPM whose deliverability is to be probed.

This abstract operation has the following results:

- a) **Submission-identifier (M)**: The probe submission identifier the MTS assigns to the probe.
- b) **Submission-time (M)**: The date and time the probe was directly submitted.

12.1.2 Originate IPM

The **Originate IPM** abstract operation originates a message whose content is an IPM.

```
originate-ipm ABSTRACT-OPERATION ::= {
  ARGUMENT SET {
    envelope [0] MessageSubmissionEnvelope,
    content  [1] IPM}
  RESULT SET {
    submission-identifier [0] MessageSubmissionIdentifier,
    submission-time       [1] MessageSubmissionTime}
  ERRORS {
    subscription-error |
    recipient-improperly-specified} }
```

This abstract operation has the following arguments:

- a) **Envelope (M)**: A message submission envelope, whose make-up the MTS Abstract Service defines. The UA supplies all but the following envelope components, which the user provides:
 - i) the desired per-message options (i.e. priority, per-message indicators, deferred delivery time, and extensions);
 - ii) the OR-names of the intended recipients and the per-recipient options (i.e. originator report request, explicit conversion, and extensions) desired for each.
- b) **Content (M)**: The IPM being originated. Its Auto-forwarded heading field shall be absent or have the value *false*.

This abstract operation has the following results:

- a) **Submission-identifier (M)**: The message submission identifier the MTS assigns to the submission.
- b) **Submission-time (M)**: The date and time the message was directly submitted.

12.1.3 Originate RN

The **Originate RN** abstract operation originates a message whose content is an RN.

```
originate-rn ABSTRACT-OPERATION ::= {
  ARGUMENT SET {
    envelope [0] MessageSubmissionEnvelope,
    content  [1] RN}
  RESULT SET {
    submission-identifier [0] MessageSubmissionIdentifier,
    submission-time      [1] MessageSubmissionTime}
  ERRORS {
    subscription-error |
    recipient-improperly-specified} }
```

An RN shall be originated only by an actual recipient of the subject IPM of whom an RN is requested by means of the Notification-requests component of the subject IPM's subject recipient specifier.

The user shall not have previously originated an RN in response to the subject IPM, by means of either the present abstract operation or auto-acknowledgment.

NOTE – If more than one copy of the same subject IPM is delivered to a recipient, then an RN may be originated for each copy received.

This abstract operation has the following arguments:

- a) **Envelope (M)**: A message submission envelope, whose make-up the MTS Abstract Service defines. The UA supplies all but the following envelope components, which the user provides:
 - i) The desired per-message options (i.e. priority, per-message indicators, and extensions). Priority shall be that of the subject IPM.
 - ii) The OR-name of the intended recipient (the Originator from the subject IPM) and the per-recipient options desired for each.
- b) **Content (M)**: The RN being originated.

This abstract operation has the following results:

- a) **Submission-identifier (M)**: The message submission identifier the MTS assigns to the submission.
- b) **Submission-time (M)**: The date and time of submission as returned by the MTA.

12.1.4 Originate ON

The **Originate ON** abstract operation originates a message whose content is an ON.

```
originate-on ABSTRACT-OPERATION ::= {
  ARGUMENT SET {
    envelope [0] MessageSubmissionEnvelope,
    content  [1] ON}
  RESULT SET {
    submission-identifier [0] MessageSubmissionIdentifier,
    submission-time      [1] MessageSubmissionTime}
  ERRORS {
    subscription-error |
    recipient-improperly-specified} }
```

There are no ONs defined in this Specification.

This abstract operation has the following arguments:

- a) **Envelope (M)**: A message submission envelope, whose make-up the MTS Abstract Service defines. The UA supplies all but the following envelope components, which the user provides:
 - i) the desired per-message options (i.e. priority, per-message indicators, and extensions);
 - ii) the OR-name of the intended recipient (the Originator from the subject IPM) and the per-recipient options desired for each.
- b) **Content (M)**: The ON being originated.

This abstract operation has the following results:

- a) **Submission-identifier (M)**: The message submission identifier the MTS assigns to the submission.
- b) **Submission-time (M)**: The date and time of submission as returned by the MTA.

12.2 Reception abstract operations

The abstract operations available at a reception port are invoked by the IPMS and performed by the user.

```
reception PORT ::= {
    SUPPLIER INVOKES { receive-report | receive-ipm | receive-rn | receive-nrn |
                      receive-on }
    ID                id-pt-reception }
```

NOTES

1 As abstractly defined, the IPMS provides no storage for received messages because whether or not it does so for a particular user has no impact upon that user's ability to communicate with other users. Thus the provision of storage is a local matter.

2 Elaborating upon the above, the *Receive IPM* abstract operation, e.g. expels an IPM from the IPMS because its purpose is to clarify the meaning of the receipt transmittal step. In contrast, the capabilities of a user to whom storage for received messages is provided might include a "Display IPM" command that enables the user to view the delivered (and perhaps already received) IPM whose IPM identifier he specifies, and that allows him to do so any number of times by repeatedly invoking the command. The first, but not subsequent uses of the command to view a particular IPM represents the concrete realization of the Receive IPM abstract operation in such an implementation.

12.2.1 Receive Report

The **Receive Report** abstract operation receives a report.

```
receive-report ABSTRACT-OPERATION ::= {
    ARGUMENT SET {
        envelope          [0] ReportDeliveryEnvelope,
        undelivered-object [1] InformationObject OPTIONAL }
```

The report received may concern any of the following previously originated by the report's recipient:

- a probe concerning a message whose content was an IPM that was originated with the Originate Probe abstract operation;
- a message whose content was an NRN that was originated as a result of *auto-discard* of *auto-forward*;
- a message whose content was an RN that was originated with the Originate RN abstract operation or by *auto-acknowledgment*;
- a message whose content was an ON that was originated with the Originate ON abstract operation;
- a message whose content was an IPM that was originated with the Originate IPM abstract operation or by *auto-forwarding*.

This abstract operation has the following arguments:

- Envelope (M):** A report delivery envelope, whose make-up the MTS Abstract Service defines.
- Undelivered-object (C):** The content of the message whose status is being reported. An IPM or IPN.

If the report was provoked by a previous Originate Probe abstract operation invocation, this conditional argument shall be absent. If the report was provoked by a previous Originate IPM abstract operation invocation, the argument shall be present if, and only if, content return was requested. Otherwise (i.e. if the report was provoked by an IPN) the argument shall be absent.

This abstract operation has no results.

12.2.2 Receive IPM

The **Receive IPM** abstract operation receives a message whose content is an IPM.

```
receive-ipm ABSTRACT-OPERATION ::= {
    ARGUMENT SET {
        envelope [0] MessageDeliveryEnvelope,
        content  [1] IPM }
```

This abstract operation has the following arguments:

- Envelope (M):** The message's delivery envelope.
- Content (M):** The IPM that is the message's content.

This abstract operation has no results.

12.2.3 Receive RN

The **Receive RN** abstract operation receives a message whose content is an RN. The RN is provoked by an IPM originated with the Originate IPM abstract operation.

```
receive-rn ABSTRACT-OPERATION ::= {
  ARGUMENT SET {
    envelope [0] MessageDeliveryEnvelope,
    content  [1] RN } }
```

This abstract operation has the following arguments:

- a) **Envelope (M)**: The message's delivery envelope.
- b) **Content (M)**: The RN that is the message's content.

This abstract operation has no results.

12.2.4 Receive NRN

The **Receive NRN** abstract operation receives a message whose content is an NRN. The NRN is provoked by an IPM originated with the Originate IPM abstract operation.

```
receive-nrn ABSTRACT-OPERATION ::= {
  ARGUMENT SET {
    envelope [0] MessageDeliveryEnvelope,
    content  [1] NRN } }
```

This abstract operation has the following arguments:

- a) **Envelope (M)**: The message's delivery envelope.
- b) **Content (M)**: The NRN that is the message's content.

This abstract operation has no results.

12.2.5 Receive ON

The **Receive ON** abstract operation receives a message whose content is an ON. The ON is provoked by an IPM originated with the Originate IPM abstract operation.

```
receive-on ABSTRACT-OPERATION ::= {
  ARGUMENT SET {
    envelope [0] MessageDeliveryEnvelope,
    content  [1] ON } }
```

This abstract operation has the following arguments:

- a) **Envelope (M)**: The message's delivery envelope.
- b) **Content (M)**: The ON that is the message's content.

This abstract operation has no results.

12.3 Management abstract operations

The abstract operations available at a management port are invoked by the user and performed by the IPMS.

```
management PORT ::= {
  CONSUMER INVOKES { change-auto-discard | change-auto-acknowledgment |
                     change-auto-forwarding }
  ID                id-pt-management }
```

12.3.1 Change Auto-discard

The **Change Auto-discard** abstract operation enables or disables **auto-discard**, the automatic discard by the IPMS of expired or obsolete IPMs delivered to, but not yet received by the user.

```
change-auto-discard ABSTRACT-OPERATION ::= {
  ARGUMENT SET {
    auto-discard-expired-IPMs [0] BOOLEAN,
    auto-discard-obsolete-IPMs [1] BOOLEAN } }
```