INTERNATIONAL TELECOMMUNICATION UNION

CCITT

THE INTERNATIONAL
TELEGRAPH AND TELEPHONE
CONSULTATIVE COMMITTEE

BLUE BOOK

VOLUME VIII – FASCICLE VIII.7

DATA COMMUNICATION NETWORKS
MESSAGE HANDLING SYSTEMS

RECOMMENDATIONS X.400-X.420

IXTH PLENARY ASSEMBLY
MELBOURNE, 14-25 NOVEMBER 1988

Geneva 1989

INTERNATIONAL TELECOMMUNICATION UNION

CCITT

THE INTERNATIONAL
TELEGRAPH AND TELEPHONE
CONSULTATIVE COMMITTEE

BLUE BOOK

VOLUME VIII – FASCICLE VIII.7

DATA COMMUNICATION NETWORKS
MESSAGE HANDLING SYSTEMS

RECOMMENDATIONS X.400-X.420

IXTH PLENARY ASSEMBLY
MELBOURNE, 14-25 NOVEMBER 1988

Geneva 1989

ISBN 92-61-03721-6

Printed in Switzerland

CONTENTS OF THE CCITT BOOK
APPLICABLE AFTER THE NINTH PLENARY ASSEMBLY (1988)

BLUE BOOK

FASCICLE III.6 – Line transmission of non-telephone signals. Transmission of sound-programme and television signals. Series H and J Recommendations (Study Group XV).

FASCICLE III.7 – Integrated Services Digital Network (ISDN) – General structure and service capabilities. Recommendations I.110-I.257 (Study Group XVIII).

FASCICLE III.8 – Integrated Services Digital Network (ISDN) – Overall network aspects and functions, ISDN user-network interfaces. Recommendations I.310-I.470 (Study Group XVIII).

FASCICLE III.9 – Integrated Services Digital Network (ISDN) – Internetwork interfaces and maintenance principles. Recommendations I.500-I.605 (Study Group XVIII).

Volume IV

FASCICLE IV.1 – General maintenance principles: maintenance of international transmission systems and telephone circuits. Recommendations M.10-M.782 (Study Group IV).

FASCICLE IV.2 – Maintenance of international telegraph, phototelegraph and leased circuits. Maintenance of the international public telephone network. Maintenance of maritime satellite and data transmission systems. Recommendations M.800-M.1375 (Study Group IV).

FASCICLE IV.3 – Maintenance of international sound-programme and television transmission circuits. Series N Recommendations (Study Group IV).

FASCICLE IV.4 – Specifications for measuring equipment. Series O Recommendations (Study Group IV).

Volume V – Telephone transmission quality. Series P Recommendations (Study Group XII).

Volume VI

FASCICLE VI.1 – General Recommendations on telephone switching and signalling. Functions and information flows for services in the ISDN. Supplements. Recommendations Q.1-Q.118 *bis* (Study Group XI).

FASCICLE VI.2 – Specifications of Signalling Systems Nos. 4 and 5. Recommendations Q.120-Q.180 (Study Group XI).

FASCICLE VI.3 – Specifications of Signalling System No. 6. Recommendations Q.251-Q.300 (Study Group XI).

FASCICLE VI.4 – Specifications of Signalling Systems R1 and R2. Recommendations Q.310-Q.490 (Study Group XI).

FASCICLE VI.5 – Digital local, transit, combined and international exchanges in integrated digital networks and mixed analogue-digital networks. Supplements. Recommendations Q.500-Q.554 (Study Group XI).

FASCICLE VI.6 – Interworking of signalling systems. Recommendations Q.601-Q.699 (Study Group XI).

FASCICLE VI.7 – Specifications of Signalling System No. 7. Recommendations Q.700-Q.716 (Study Group XI).

FASCICLE VI.8 – Specifications of Signalling System No. 7. Recommendations Q.721-Q.766 (Study Group XI).

FASCICLE VI.9 – Specifications of Signalling System No. 7. Recommendations Q.771-Q.795 (Study Group XI).

FASCICLE VI.10 – Digital subscriber signalling system No. 1 (DSS 1), data link layer. Recommendations Q.920-Q.921 (Study Group XI).

FASCICLE VI.11 – Digital subscriber signalling system No. 1 (DSS 1), network layer, user-network management. Recommendations Q.930-Q.940 (Study Group XI).

FASCICLE VI.12 – Public land mobile network. Interworking with ISDN and PSTN. Recommendations Q.1000-Q.1032 (Study Group XI).

FASCICLE VI.13 – Public land mobile network. Mobile application part and interfaces. Recommendations Q.1051-Q.1063 (Study Group XI).

FASCICLE VI.14 – Interworking with satellite mobile systems. Recommendations Q.1100-Q.1152 (Study Group XI).

Volume VII

FASCICLE VII.1 – Telegraph transmission. Series R Recommendations. Telegraph services terminal equipment. Series S Recommendations (Study Group IX).

FASCICLE VII.2 – Telegraph switching. Series U Recommendations (Study Group IX).

FASCICLE VII.3 – Terminal equipment and protocols for telematic services. Recommendations T.0-T.63 (Study Group VIII).

FASCICLE VII.4 – Conformance testing procedures for the Teletex Recommendations. Recommendation T.64 (Study Group VIII).

FASCICLE VII.5 – Terminal equipment and protocols for telematic services. Recommendations T.65-T.101, T.150-T.390 (Study Group VIII).

FASCICLE VII.6 – Terminal equipment and protocols for telematic services. Recommendations T.400-T.418 (Study Group VIII).

FASCICLE VII.7 – Terminal equipment and protocols for telematic services. Recommendations T.431-T.564 (Study Group VIII).

Volume VIII

FASCICLE VIII.1 – Data communication over the telephone network. Series V Recommendations (Study Group XVII).

FASCICLE VIII.2 – Data communication networks: services and facilities, interfaces. Recommendations X.1-X.32 (Study Group VII).

FASCICLE VIII.3 – Data communication networks: transmission, signalling and switching, network aspects, maintenance and administrative arrangements. Recommendations X.40-X.181 (Study Group VII).

FASCICLE VIII.4 – Data communication networks: Open Systems Interconnection (OSI) – Model and notation, service definition. Recommendations X.200-X.219 (Study Group VII).

FASCICLE VIII.5 – Data communication networks: Open Systems Interconnection (OSI) – Protocol specifications, conformance testing. Recommendations X.220-X.290 (Study Group VII).

FASCICLE VIII.6 – Data communication networks: interworking between networks, mobile data transmission systems, internetwork management. Recommendations X.300-X.370 (Study Group VII).

FASCICLE VIII.7 – Data communication networks: message handling systems. Recommendations X.400-X.420 (Study Group VII).

FASCICLE VIII.8 – Data communication networks: directory. Recommendations X.500-X.521 (Study Group VII).

Volume IX – Protection against interference. Series K Recommendations (Study Group V). Construction, installation and protection of cable and other elements of outside plant. Series L Recommendations (Study Group VI).

Volume X

FASCICLE X.1 – Functional Specification and Description Language (SDL). Criteria for using Formal Description Techniques (FDTs). Recommendation Z.100 and Annexes A, B, C and E, Recommendation Z.110 (Study Group X).

FASCICLE X.2 – Annex D to Recommendation Z.100: SDL user guidelines (Study Group X).

FASCICLE X.3 – Annex F.1 to Recommendation Z.100: SDL formal definition. Introduction (Study Group X).

FASCICLE X.4 – Annex F.2 to Recommendation Z.100: SDL formal definition. Static semantics (Study Group X).

FASCICLE X.5 – Annex F.3 to Recommendation Z.100: SDL formal definition. Dynamic semantics (Study Group X).

FASCICLE X.6 – CCITT High Level Language (CHILL). Recommendation Z.200 (Study Group X).

FASCICLE X.7 – Man-Machine Language (MML). Recommendations Z.301-Z.341 (Study Group X).

CONTENTS OF FASCICLE VIII.7 OF THE BLUE BOOK

Recommendations X.400 to X.420

Data communication networks:
Message handling systems (MHS)

PRELIMINARY NOTES

1 The Questions entrusted to each Study Group for the Study Period 1989-1992 can be found in Contribution No. 1 to that Study Group.

2 In this fascicle, the expression "Administration" is used for shortness to indicate both a telecommunication Administration and a recognized private operating agency.

3 The status of annexes and appendices attached to the Series X Recommendations should be interpreted as follows (except where otherwise specified):

- an *annex* to a Recommendation forms an integral part of the Recommendation;
- an *appendix* to a Recommendation does not form part of the Recommendation and only provides some complementary explanation or information specific to that Recommendation.

4 A number of the Series X Recommendations contained in this Fascicle were jointly developed in collaboration with the ISO/IEC. Cross-references between these Recommendations and the corresponding ISO/IEC standards are given in the table below.

CCITT Recommendation	ISO/IEC Standard
X.400	ISO 10021-1, Information processing systems – Text communication – Message oriented text interchange system – Part 1: System and service overview [a].
X.402	ISO 10021-2, Information processing systems – Text communication – Message oriented text interchange system – Part 2: Overall architecture [a].
X.407	ISO 10021-3, Information processing systems – Text communication – Message oriented text interchange system – Part 3: Abstract service definition conventions [a].
X.411	ISO 10021-4, Information processing systems – Text communication – Message oriented text interchange system – Part 4: Message transfer system: Abstract service definition and procedures [a].
X.413	ISO 10021-5, Information processing systems – Text communication – Message oriented text interchange system – Part 5: Message store: Abstract service definition. [a]
X.419	ISO 10021-6, Information processing systems – Text communication – Message oriented text interchange system – Part 6: Protocol specifications [a].
X.420	ISO 10021-7, Information processing systems – Text communication – Message oriented text interchange system – Part 7: Interpersonal messaging system [a].

[a] Presently at the stage of Draft International Standard (DIS).

FASCICLE VIII.7

Recommendations X.400 to X.420

DATA COMMUNICATION NETWORKS: MESSAGE HANDLING SYSTEMS

PAGE INTENTIONALLY LEFT BLANK

PAGE LAISSEE EN BLANC INTENTIONNELLEMENT

Recommendation X.400 [1]

MESSAGE HANDLING SYSTEM AND SERVICE OVERVIEW

The establishment in various countries of telematic services and computer based store and forward messaging services in association with public networks creates a need to produce standards to facilitate international message exchange between subscribers to such services.

The CCITT,

considering

(a) the need for message handling systems;

(b) the need to transfer and store messages of different types;

(c) that Recommendation X.200 defines the reference model of open systems interconnection for CCITT applications;

(d) that Recommendations X.208, X.217, X.218 and X.219 provide the foundation for CCITT applications;

(e) that the X.500-Series Recommendations define directory systems;

(f) that message handling systems are defined in a series of Recommendations: X.400, X.402, X.403, X.407, X.408, X.411, X.413 and X.419;

(g) that interpersonal message is defined in Recommendation X.420 and T.330;

(h) that several F-Series Recommendations describe public message handling services: F.400, F.401, F.410 and F.420;

(i) that several F-Series Recommendations describe intercommunication between public message handling services and other services: F.421, F.415 and F.422,

unanimously declares

the view that the overall system and service overview of message handling is defined in this Recommendation.

---

[1] Recommendation F.400 is identical to X.400.

CONTENTS

PART 1 – INTRODUCTION

0 Introduction

This Recommendation is one of a set of Recommendations for message handling. The entire set provides a comprehensive specification for message handling comprising any number of cooperating open-systems.

Message handling systems and services enable users to exchange messages on a store-and-forward basis. A message submitted by one user, the originator, is conveyed by the message transfer system (MTS), the principal component of a larger message handling system (MHS), and is subsequently delivered to one or more additional users, the message's recipients.

An MHS comprises a variety of interconnected functional entities. Message transfer agents (MTAs) cooperate to perform the store-and-forward message transfer function. Message stores (MSs) provide storage for messages and enable their submission, retrieval and management. User agents (UAs) help users access MHS. Access units (AUs) provide links to other communication systems and services of various kinds (e.g. other telematic services, postal services).

This Recommendation specifies the overall system and service description of message handling capabilities.

1 Scope

This Recommendation defines the overall system and service of an MHS and serves as a general overview of MHS.

Other aspects of message handling systems and services are defined in other Recommendations. The layout of Recommendations defining the message handling system and services is shown in Table 1/X.400. The public services built on MHS, as well as access to and from the MHS for public services are defined in the F.400-Series of Recommendations.

The technical aspects of MHS are defined in the X.400-Series of Recommendations. The overall system architecture of MHS is defined in Recommendation X.402.

TABLE 1/X.400

Structure of MHS Recommendations

Name of Recommendation/Standard	Joint MHS		Joint support		CCITT only	
	CCITT	ISO	CCITT	ISO	System	Service
MHS: System and service overview	X.400	10021-1				F.400
MHS: Overall architecture	X.402	10021-2				
MHS: Conformance testing					X.403	
MHS: Abstract service definition conventions	X.407	10021-3				
MHS: Encoded information type conversion rules					X.408	
MHS: MTS: Abstract service definition and procedures	X.411	10021-4				
MHS: MS: Abstract service definition	X.413	10021-5				
MHS: Protocol specifications	X.419	10021-6				
MHS: Interpersonal messaging system	X.420	10021-7				
Telematic access to IPMS					T.330	
MHS: Naming and addressing for public MH services						F.401
MHS: The public message transfer service						F.410
MHS: Intercommunication with public physical delivery services						F.415
MHS: The public IPM service						F.420
MHS: Intercommunication between IPM service and telex						F.421
MHS: Intercommunication between IPM service and teletex						F.422
OSI: Basic reference model			X.200	7498		
OSI: Specification of abstract syntax notation one (ASN.1)			X.208	8824		
OSI: Specification of basic encoding rules for abstract syntax notation one (ASN.1)			X.209	8825		
OSI: Association control: service definition			X.217	8649		
OSI: Reliable transfer: model and service definition			X.218	9066-1		
OSI: Remote operations: model, notation and service definition			X.219	9072-1		
OSI: Association control: protocol specification			X.227	8650		
OSI: Reliable transfer: protocol specification			X.228	9066-2		
OSI: Remote operations: protocol specification			X.229	9072-2		

2 References

This Recommendation cites the documents listed below:

Recommendation F.60 Operational provisions for the international telex service

Recommendation F.69 Plan for the telex destination codes

Recommendation F.72 International telex store-and-forward – General principles and operational aspects

Recommendation F.160 General operational provisions for the international public fascimile services

Recommendation F.200 Teletex service

Recommendation F.300 Videotex service

Recommendation F.400 Message handling – System and service overview (see also ISO 10021-1)

Recommendation F.401 Message handling services – Naming and addressing for public message handling services

Recommendation F.410 Message handling services – The public message transfer service

Recommendation F.415 Message handling services – Intercommunication with public physical delivery services

Recommendation F.420 Message handling services – The public interpersonal messaging service

Recommendation F.421 Message handling services – Intercommunication between the IPM service and the telex service

Recommendation F.422 Message handling services – Intercommunication between the IPM service and the teletex service

Recommendation T.61 Character repertoire and coded character sets for the international teletex service

Recommendation T.330 Telematic access to IPMS

Recommendation U.80 International teletex store-and-forward – Access from telex

Recommendation U.204 Interworking between the telex service and the public interpersonal messaging service

Recommendation X.200 Reference model of open systems interconnection for CCITT applications (see also ISO 7498)

Recommendation X.208 Specification of abstract syntax notation one (ASN.1) (see also ISO 8824)

Recommendation X.209 Specification of basic encoding rules for abstract syntax notation one (ASN.1) (see also ISO 8825)

Recommendation X.217 Association control: Service definitions (see also ISO 8649)

Recommendation X.218 Reliable transfer model: Service definition (see also ISO/IEC 9066-1)

Recommendation X.219 Remote operations model: Notation and service definition (see also ISO/IEC 9072-1)

Recommendation X.400 Message handling – System and service overview (see also ISO/IEC 10021-1)

Recommendation X.402 Message handling systems – Overall architecture (see also ISO/IEC 10021-2)

Recommendation X.403 Message handling systems – Conformance testing

Recommendation X.407 Message handling systems – Abstract service definition conventions (see also ISO/IEC 10021-3)

Recommendation X.408 Message handling systems – Encoded information type convention rules

Recommendation X.411 Message handling systems – Message transfer system: Abstract service definition and procedures (see also ISO/IEC 10021-4)

Recommendation X.413	Message handling systems – Message store: Abstract service definition (see also ISO/IEC 10021-5)
Recommendation X.419	Message handling systems – Protocol specifications (see also ISO/IEC 10021-6)
Recommendation X.420	Message handling systems – Interpersonal messaging system (see also ISO/IEC 10021-7)
Recommendation X.500	Directory – Overview (see also ISO/IEC 9594-1)
Recommendation X.501	Directory – Models (see also ISO/IEC 9594-2)
Recommendation X.509	Directory – Authentication (see also ISO/IEC 9594-8)
Recommendation X.511	Directory – Abstract service definition (see also ISO/IEC 9594-3)
Recommendation X.518	Directory – Procedures for distributed operations (see also ISO/IEC 9594-4)
Recommendation X.519	Directory – Protocol specifications (see also ISO/IEC 9594-5)
Recommendation X.520	Directory – Selected attribute types (see also ISO/IEC 9594-6)
Recommendation X.521	Directory – Selected object classes (see also ISO/IEC 9594-7)

3 Definitions

This Recommendation uses the terms listed below, and those defined in Annex A.

Definitions of the elements of service applicable to MHS are contained in Annex B.

3.1 *Open systems interconnection*

This Recommendation uses the following terms defined in Recommendation X.200:

a) Application layer;
b) Application-process;
c) Open systems interconnection;
d) OSI reference model.

3.2 *Directory systems*

This Recommendation uses the following terms defined in Recommendation X.500:

a) directory entry;
b) directory system agent;
c) directory system;
d) directory user agent.

This Recommendation uses the following terms defined in Recommendation X.501:

e) attribute;
f) group;
g) member;
h) name.

4 Abbreviations

A	Additional
ADMD	Administration management domain
AU	Access unit
CA	Contractual agreement
DL	Distribution list
DSA	Directory system agent
DUA	Directory user agent

E	Essential
EIT	Encoded information type
I/O	Input/output
IP	Interpersonal
IPM	Interpersonal messaging
IPMS	Interpersonal messaging system
MD	Management domain
MH	Message handling
MHS	Message handling system
MS	Message store
MT	Message transfer
MTA	Message transfer agent
MTS	Message transfer system
N/A	Not applicable
O/R	Originator/recipient
OSI	Open system interconnection
PD	Physical delivery
PDAU	Physical delivery access unit
PDS	Physical delivery system
PM	Per-message
PR	Per-recipient
PRMD	Private management domain
PTLXAU	Public telex access unit
TLMA	Telematic agent
TLXAU	Telex access unit
TTX	Teletex
UA	User agent

5 Conventions

In this Recommendation the expression "Administration" is used for shortness to indicate a telecommunication Administration, a recognized private operating agency, and, in the case of intercommunication with public delivery service, a postal Administration.

Note – This Recommendation is identical to Recommendation F.400. Because of the desired alignment with ISO, the conventions of ISO standards have been adopted for the structure of this text. These conventions differ from the CCITT style. The other Recommendations of the X.400-Series are in accordance with CCITT conventions.

PART 2 – GENERAL DESCRIPTION OF MHS

6 Purpose

This Recommendation is one of a set of Recommendations and describes the system model and elements of service of the message handling system (MHS) and services. This Recommendation overviews the capabilities of an MHS that are used by Administrations for the provision of public MH services to enable users to exchange messages on a store-and-forward basis.

The message handling system is designed in accordance with the principles of the reference model of open systems interconnection (OSI reference model) for CCITT applications (Recommendation X.200) and uses the presentation layer services and services offered by other, more general, application service elements. An MHS can be constructed using any network fitting in the scope of OSI. The message transfer service provided by the MTS is application independent. An example of a standardized application is the IPM service. End systems can use the MT service for specific applications that are defined bilaterally.

Message handling services provided by Administrations belong to the group of telematic services defined in F-Series Recommendations.

Various other telematic services and telex (Recommendations F.60, F.160, F.200, F.300, etc.), data transmission services (X.1), or physical delivery services (F.415) gain access to, and intercommunicate with, the IPM service or intercommunicate with each other, via access units.

Elements of service are the service features provided through the application processes. The elements of service are considered to be components of the services provided to users and are either elements of a basic service or they are *optional user facilities*, classified either as *essential optional user facilities*, or as *additional optional user facilities*.

7 Functional model of MHS

The MHS functional model serves as a tool to aid in the development of Recommendations for MHS, and aids in describing the basic concepts that can be depicted graphically. It comprises several different functional components that work together to provide MH services. The model can be applied to a number of different physical and organizational configurations.

7.1 *Description of the MHS model*

A functional view of the MHS model is shown in Figure 1/X.400. In this model, a user is either a person or a computer process. Users are either direct users (i.e. engage in message handling by direct use of MHS), or are indirect users (i.e. engage in message handling through another communication system (e.g. a physical delivery system) that is linked to MHS). A user is referred to as either an originator (when sending a message) or a recipient (when receiving a message). Message handling elements of service define the set of message types and the capabilities that enable an originator to transfer messages of those types to one or more recipients.

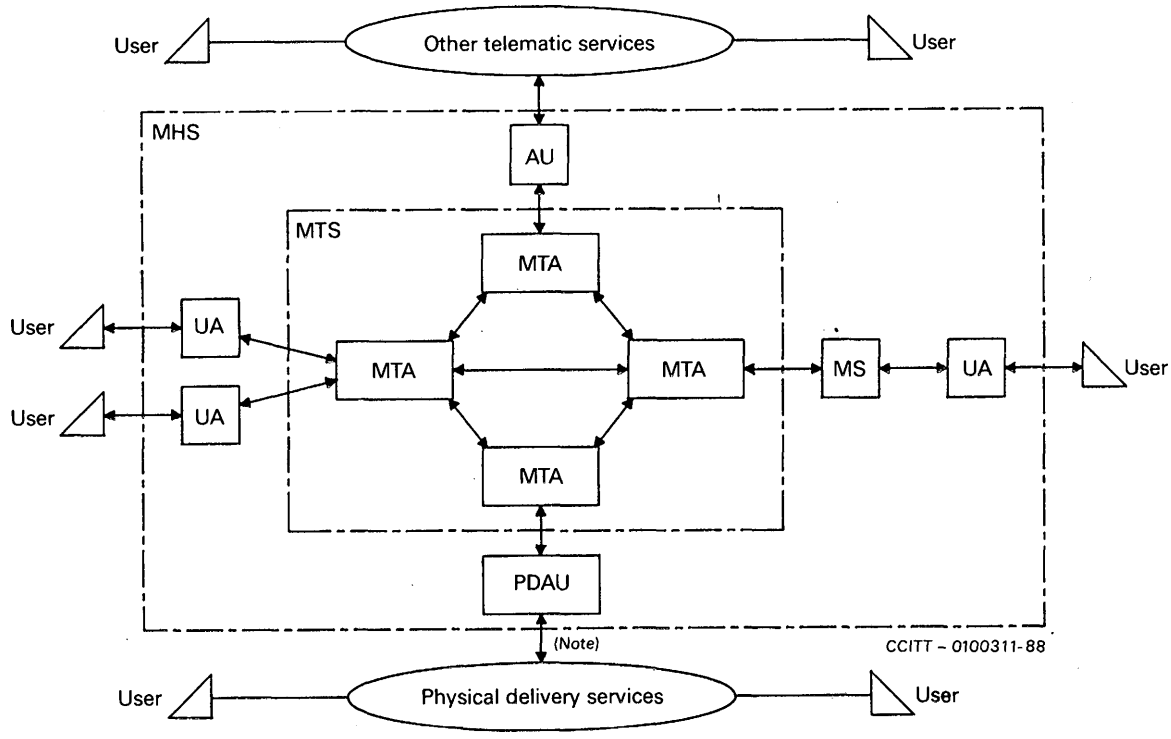
An originator prepares messages with the assistance of his user agent. A user agent (UA) is an application process that interacts with the message transfer system (MTS) or a message store (MS), to submit messages on behalf of a single user. The MTS delivers the messages submitted to it, to one or more recipient UAs, access units (AUs), or MSs, and can return notifications to the originator. Functions performed solely by the UA and not standardized as part of the message handling elements of service are called local functions. A UA can accept delivery of messages directly from the MTS, or it can use the capabilities of an MS to receive delivered messages for subsequent retrieval by the UA.

The MTS comprises a number of message transfer agents (MTAs). Operating together, in a store-and-forward manner, the MTAs transfer messages and deliver them to the intended recipients.

Access by indirect users of MHS is accomplished by AUs. Delivery to indirect users of MHS is accomplished by AUs, such as in the case of physical delivery, by the physical delivery access unit (PDAU).

The message store (MS) is an optional general purpose capability of MHS that acts as an intermediary between the UA and the MTA. The MS is depicted in the MHS functional model shown in Figure 1/X.400. The MS is a functional entity whose primary purpose is to store and permit retrieval of delivered messages. The MS also allows for submission from, and alerting to the UA.

The collection of UAs, MSs, AUs and MTAs is called the message handling system (MHS).



Note – Message input from PD Services to MHS is for further study. Flow from PD Services to the PDAU shown is for notifications.

FIGURE 1/X.400

MHS functional model

7.2 *Structure of messages*

The basic structure of messages conveyed by the MTS is shown in Figure 2/X.400. A message is made up of an envelope and a content. The envelope carries information that is used by the MTS when transferring the message within the MTS. The content is the piece of information that the originating UA wishes delivered to one or more recipient UAs. The MTS neither modifies or examines the content, except for conversion (see § 16).

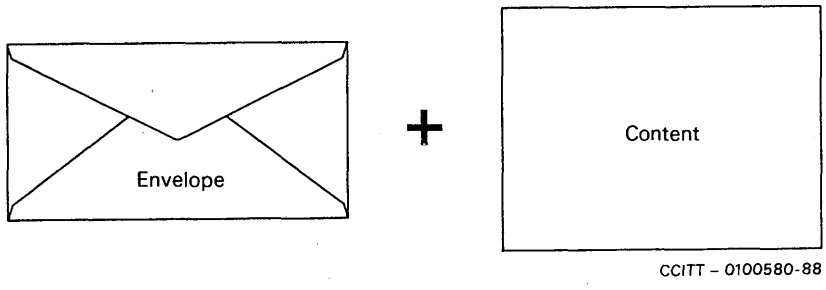


FIGURE 2/X.400

Basic message structure

7.3 . *Application of the MHS model*

7.3.1 *Physical mapping*

Users access UAs for message processing purposes, for example, to create, present, or file messages. A user can interact with a UA via an input/output device or process (e.g. keyboard, display, printer, etc.). A UA can be implemented as a (set of) computer process(es) in an intelligent terminal.

A UA and MTA can be co-located in the same system, or a UA/MS can be implemented in physically separate systems. In the first case the UA accesses the MT elements of service by interacting directly with the MTA in the same system. In the second case, the UA/MS will communicate with the MTA via standardized protocols specified for MHS. It is also possible for an MTA to be implemented in a system without UAs or MSs.

Some possible physical configurations are shown in Figures 3/X.400 and 4/X.400. The different physical systems can be connected by means of dedicated lines or switched network connections.

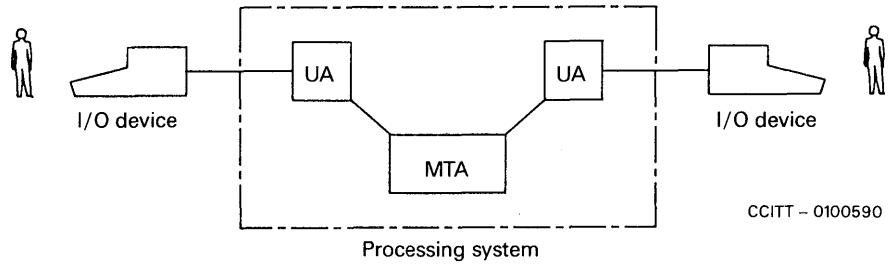


FIGURE 3/X.400

Co-resident UA and MTA

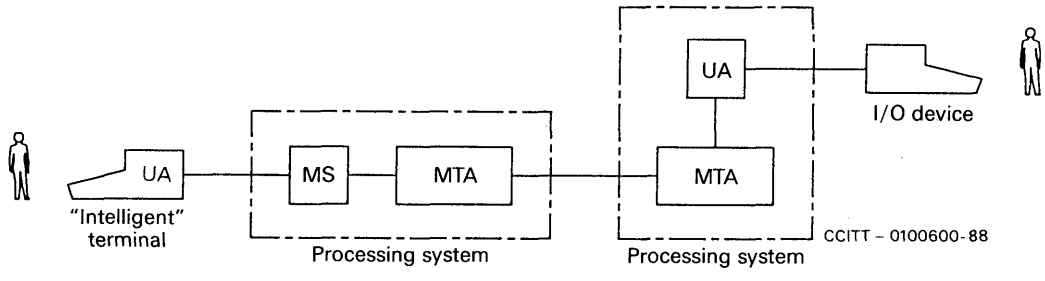


FIGURE 4/X.400

Stand-alone UA and co-resident MS/MTA and US/MTA

7.3.2 *Organizational mapping*

An Administration or organization can play various roles in providing message handling services. An organization in this context can be a company or a non-commercial enterprise.

The collection of at least one MTA, zero or more UAs, zero or more MSs, and zero or more AUs operated by an Administration or organization constitutes a management domain (MD). An MD managed by an Administration is called an Administration management domain (ADMD). An MD managed by an organization other than an Administration is called a private management domain (PRMD). An MD provides message handling services in accordance with the classification of elements of service as described in § 19. The relationships between management domains is shown in Figure 5/X.400.

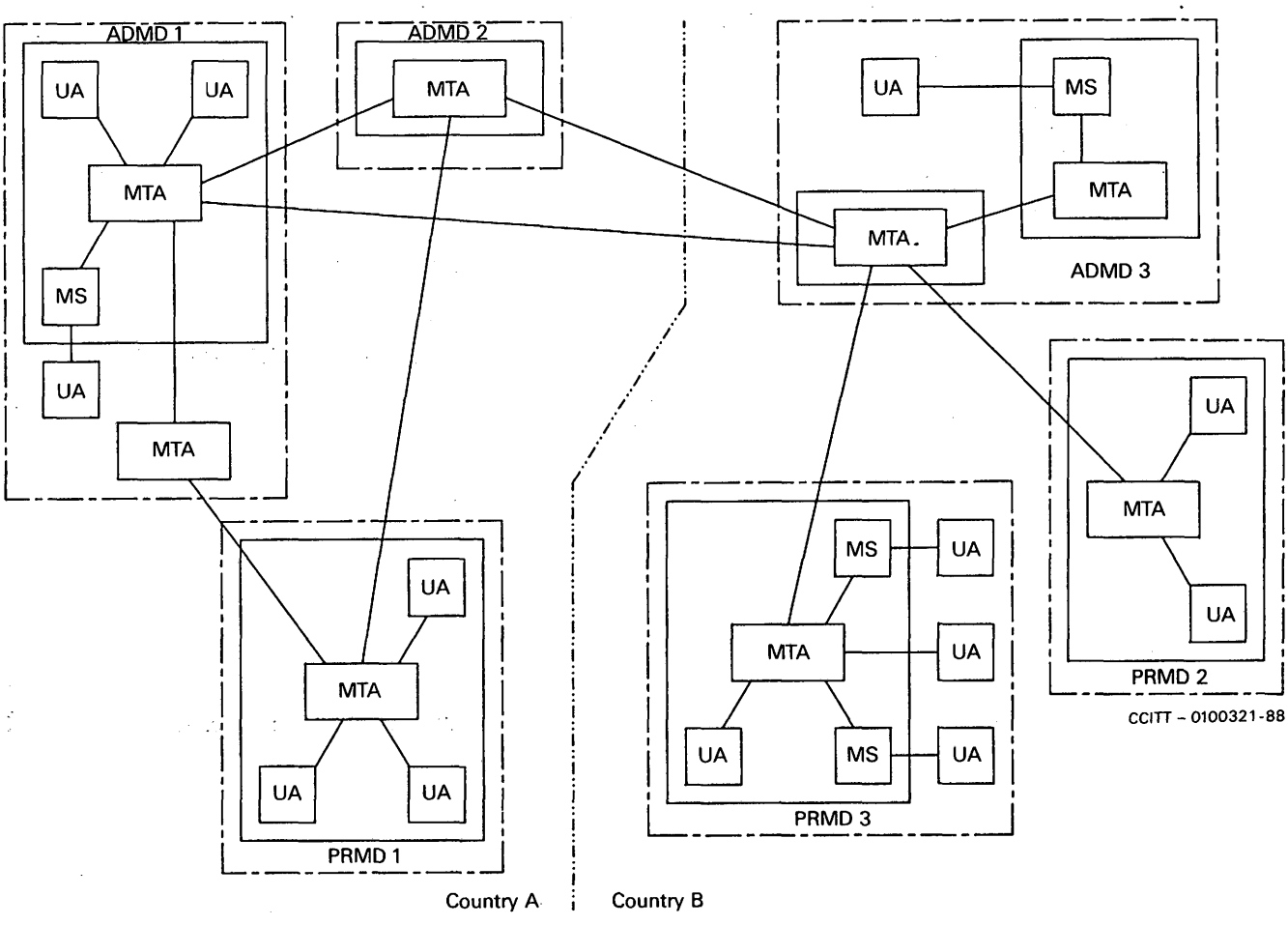


FIGURE 5/X.400

Relationships between management domains

Note 1 – It should be recognized that the provision of support of private messaging systems by CCITT members falls within the framework of national regulations. Thus the possibilities mentioned in this paragraph may or may not be offered by an Administration which provides message handling services. In addition, the UAs depicted in Figure 5/X.400 do not imply that UAs belonging to an MD must be exclusively located in the same country as their MDs.

Note 2 – Direct interactions between PRMDs and internal interactions within an MD are outside the scope of this Recommendation.

Note 3 – An Administration, in the context of CCITT, that manages an ADMD, is understood as being a member of ITU or a recognized private operating agency (RPOA), registered by a country with the ITU.

7.3.3 *Administration management domain*

In one country one or more ADMDs can exist. An ADMD is characterized by its provision of relaying functions between other management domains and the provision of message transfer service for the applications provided within the ADMD.

An Administration can provide access for its users to the ADMD in one or more of the following ways:

- users to Administration provided UA
- private UA to Administration MTA
- private UA to Administration MS
- private UA to Administration MTA
- user to Administration provided UA.

See also the examples of configurations shown in Figure 3/X.400 and Figure 4/X.400.

Administration provided UAs can exist as part of an intelligent terminal that the user can use to access MHS. They can also exist as part of Administration resident equipment being part of MHS, in which case the user obtains access to the UA via an I/O device.

In the case of a private UA, the user has a private stand-alone UA which interacts with the Administration provided MTA or MS, using submission, delivery and retrieval functions. A private, stand-alone UA can be associated with one or more MDs, provided that the required naming conventions are preserved.

A private MTA as part of an PRMD can access one or more ADMDs in a country, following national regulations.

Access can also be provided by Administration provided AUs described in §§ 10 and 11.

7.3.4 *Private management domain*

An organization other than an Administration can have one or more MTA(s), and zero or more UAs, AUs and MSs forming a PRMD which can interact with an ADMD on an MD to MD (MTA to MTA) basis. A PRMD is characterized by the provision of messaging functions within that management domain.

A PRMD is considered to exist entirely within one country. Within that country, the PRMD can have access to one or more ADMDs as shown in Figure 5/X.400. However, in the case of a specific interaction between a PRMD and an ADMD (such as when a message is transferred between MDs), the PRMD is considered to be associated only with that ADMD. A PRMD will not act as a relay between two ADMDs.

In the interaction between a PRMD and an ADMD, the ADMD takes responsibility for the actions of the PRMD which are related to the interaction. In addition to ensuring that the PRMD properly provides the message transfer service, the ADMD is responsible for ensuring that the accounting, logging, quality of service, uniqueness of names, and related operations of the PRMD are correctly performed. As a national matter, the name of a PRMD can be either nationally unique or relative to the associated ADMD. If a PRMD is associated with more than one ADMD, the PRMD can have more than one name.

7.4 *Message store*

Because UAs can be implemented on a wide variety of equipment, including personal computers, the MS can complement a UA implemented, for example, on a personal computer by providing a more secure, continuously available storage mechanism to take delivery of messages on the user agent's behalf. The MS retrieval capability provides users who subscribe to an MS with basic message retrieval capabilities potentially applicable to messages of all types. Figure 6/X.400 shows the delivery, and subsequent retrieval of messages that are delivered to an MS, and the indirect submission of messages via the MS.

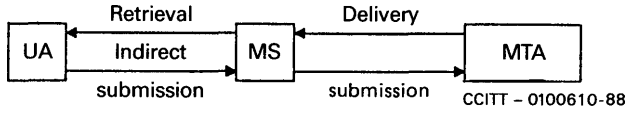


FIGURE 6/X.400

Submission and delivery with an MS

One MS acts on behalf of only one user (one O/R address), i.e. it does not provide a common or shared MS capability to several users (see also PRMD3 of Figure 5/X.400).

When subscribing to an MS, all messages destined for the UA are delivered to the MS only. The UA, if on line, can receive alerts when certain messages are delivered to the MS. Messages delivered to an MS are considered delivered from the MTS perspective.

When a UA submits a message through the MS, the MS is in general transparent and submits it to the MTA before confirming the success of the submission to the UA. However, the MS can expand the message if the UA requests the forwarding of messages that exist in the MS.

Users are also provided with the capability to request the MS to forward selected messages automatically upon delivery.

The elements of service describing the features of the MS are defined in Annex B and classified in § 19. Users are provided with the capability based on various criteria, to get counts and lists of messages, to fetch messages, and to delete messages, currently held in the MS.

7.4.1 *Physical configurations*

The MS can be physically located with respect to the MTA in a number of ways. The MS can be co-located with the UA, co-located with the MTA, or stand-alone. From an external point of view, a co-located UA and MS are indistinguishable from a stand-alone UA. Co-locating the MS with the MTA offers significant advantages which will probably make it the predominant configuration.

7.4.2 *Organizational configurations*

Either ADMDs or PRMDs can operate MSs. In the case of Administration supplied MSs, the subscriber either provides his own UA or makes use of an Administration supplied UA via an I/O device. In either case, all the subscriber's messages are delivered to the MS for subsequent retrieval.

The physical and organizational configurations described above are examples only and other equally cases can exist.

8 Message transfer service

The MTS provides the general, application independent, store-and-forward message transfer service. The elements of service describing the features of the MT service are defined in Annex B and classified in § 19. Provision of public message transfer service by Administrations is described in Recommendation F.410.

8.1 *Submission and delivery*

The MTS provides the means by which UAs exchange messages. There are two basic interactions between MTAs and UAs and/or MSs:

1) The submission interaction is the means by which an originating UA or MS transfers to an MTA the content of a message and the submission envelope. The submission envelope contains the information that the MTS requires to provide the requested elements of service.
2) The delivery interaction is the means by which the MTA transfers to a recipient UA or MS the content of a message plus the delivery envelope. The delivery envelope contains information related to delivery of the message.

In the submission and delivery interactions, responsibility for the message is passed between the MTA and the UA or MS.

8.2 *Transfer*

Starting at the originator's MTA, each MTA transfers the message to another MTA until the message reaches the recipient's MTA, which then delivers it to the recipient UA or MS using the delivery interaction.

The transfer interaction is the means by which one MTA transfers to another MTA the content of a message plus the transfer envelope. The transfer envelope contains the information related to the operation of the MTS plus information that the MTS requires to provide elements of service requested by the originating UA.

MTAs transfer messages containing any type of binary coded information. MTAs neither interpret not alter the content of messages except when performing a conversion.

8.3 *Notifications*

Notifications in the MT service comprise the delivery and non-delivery notifications. When a message, or probe, cannot be delivered by the MTS, a non-delivery notification is generated and returned to the originator in a report signifying this. In addition, an originator can specifically ask for acknowledgement of successful delivery through use of the delivery notification element of service on submission.

8.4 *User agent*

The UA uses the MT service provided by the MTS. A UA is a functional entity by means of which a single direct user engages in message handling.

UAs are grouped into classes based on the type of content of messages they can handle. The MTS provides a UA with the ability to identify its class when sending messages to other UAs. UAs within a given class are referred to as cooperating UAs since they cooperate with each other to enhance the communication amongst their respective users.

Note – A UA can support more than one type of message content, and hence belong to several UA classes.

8.5 *Message store*

The message store (MS) uses the MT service provided by the MTS. An MS is a functional entity associated with a user's UA. The user can submit messages through it, and retrieve messages that have been delivered to the MS.

8.6 *Access unit*

An access unit (AU) uses the MT service provided by the MTS. An AU is a functional entity associated with an MTA to provide for intercommunication between MHS and another system or service.

8.7 *Use of the MTS in the provision of various services*

The MTS is used by application specific services for the provision of message handling services of various types. The interpersonal messaging service, described in § 9, is one example of this. Other services can be built on the foundation of the MTS, either with corresponding recommendations or as private applications.

9 IPM service

The interpersonal message service (IPM service) provides a user with features to assist in communicating with other IPM service users. The IPM service uses the capabilities of the MT service for sending and receiving interpersonal messages. The elements of service describing the features of the IPM service are defined in Annex B and classified in § 19. The provision of public interpersonal messaging service by Administrations is described in Recommendation F.420.

9.1 *IPM service functional model*

Figure 7/X.400 shows the functional model of the IPM service. The UAs used in the IPM service (IPM-UAs) comprise a specific class of cooperating UAs. The optional access units shown (TLMA, PTLXAU) allow for teletex and telex users to intercommunicate with the IPM service. The optional access unit (TLMA) also allows for teletex users to participate in the IPM service (see also § 11). The optional physical delivery access unit (PDAU) allows IPM users to send messages to users outside the IPM service who have no access to MHS. The message store can optionally be used by IPM users to take delivery of messages on their behalf.

9.2 *Structure of IP-messages*

The IP class of UAs create messages containing a content specific to the IPM. The specific content that is sent from one IPM UA to another is a result of an originator composing and sending a message, called an IP-message. The structure of an IP-message as it release to the basic message structure of MHS is shown in Figure 8/X.400. The IP-message is conveyed with an envelope when being transferred through the MTS.

Figure 9/X.400 shows an analogy between a typical office memo, and the corresponding IP-message structure. The IP-message contains information (e.g., to, cc, subject) provided by the user which is transformed by the IPM UA into the heading of the IP-message. The main information that the user wishes to communicate (the body of the memo) is contained within the body of the IP-message. In the example shown, the body contains two types of encoded information: text and facsimile, which form what are called body parts. In general, an IP-message body can consist of a number of body parts, each which can be of a different encoded information type, such as voice, text, facsimile and graphics.

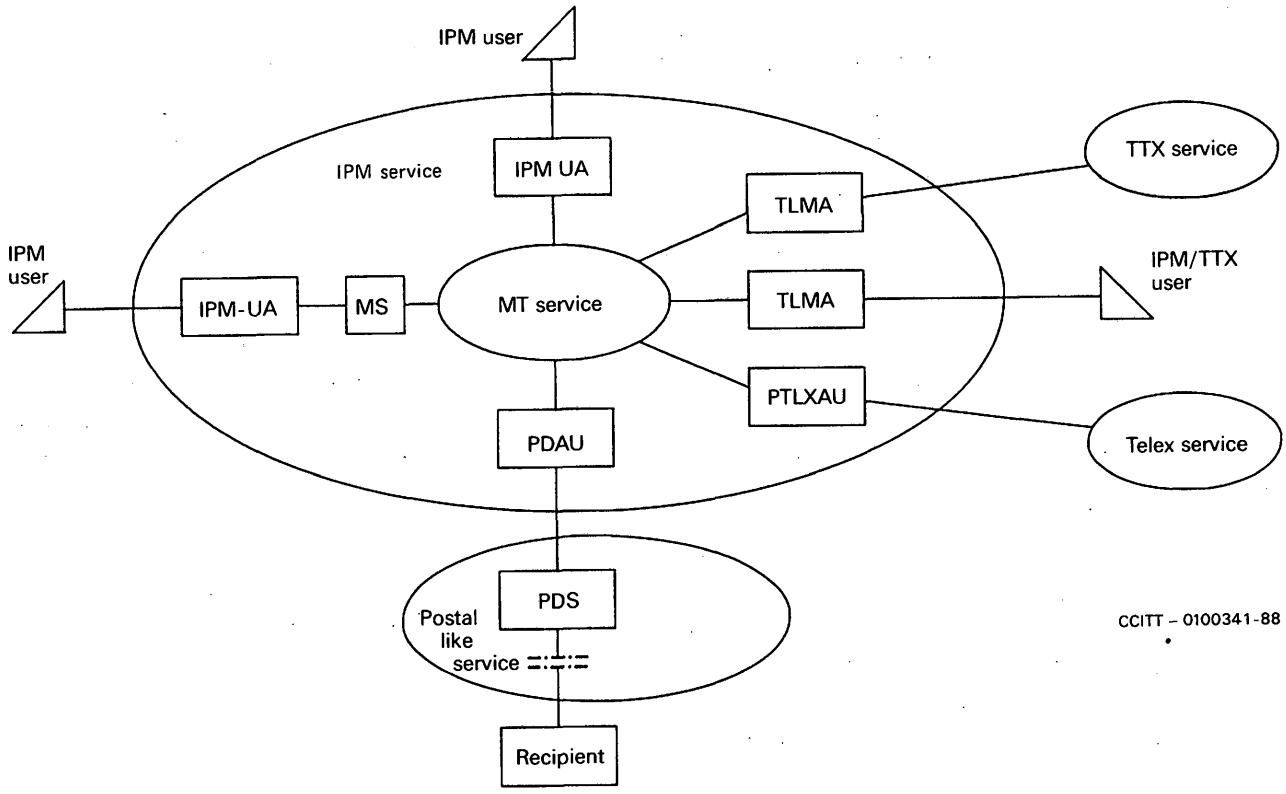


FIGURE 7/X.400

IPM service functional model

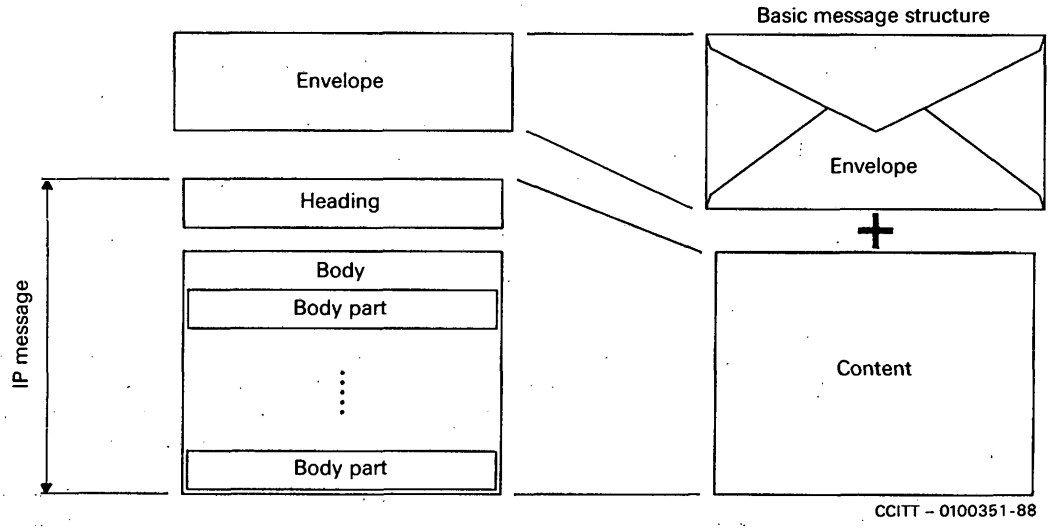


FIGURE 8/X.400

IP-message structure

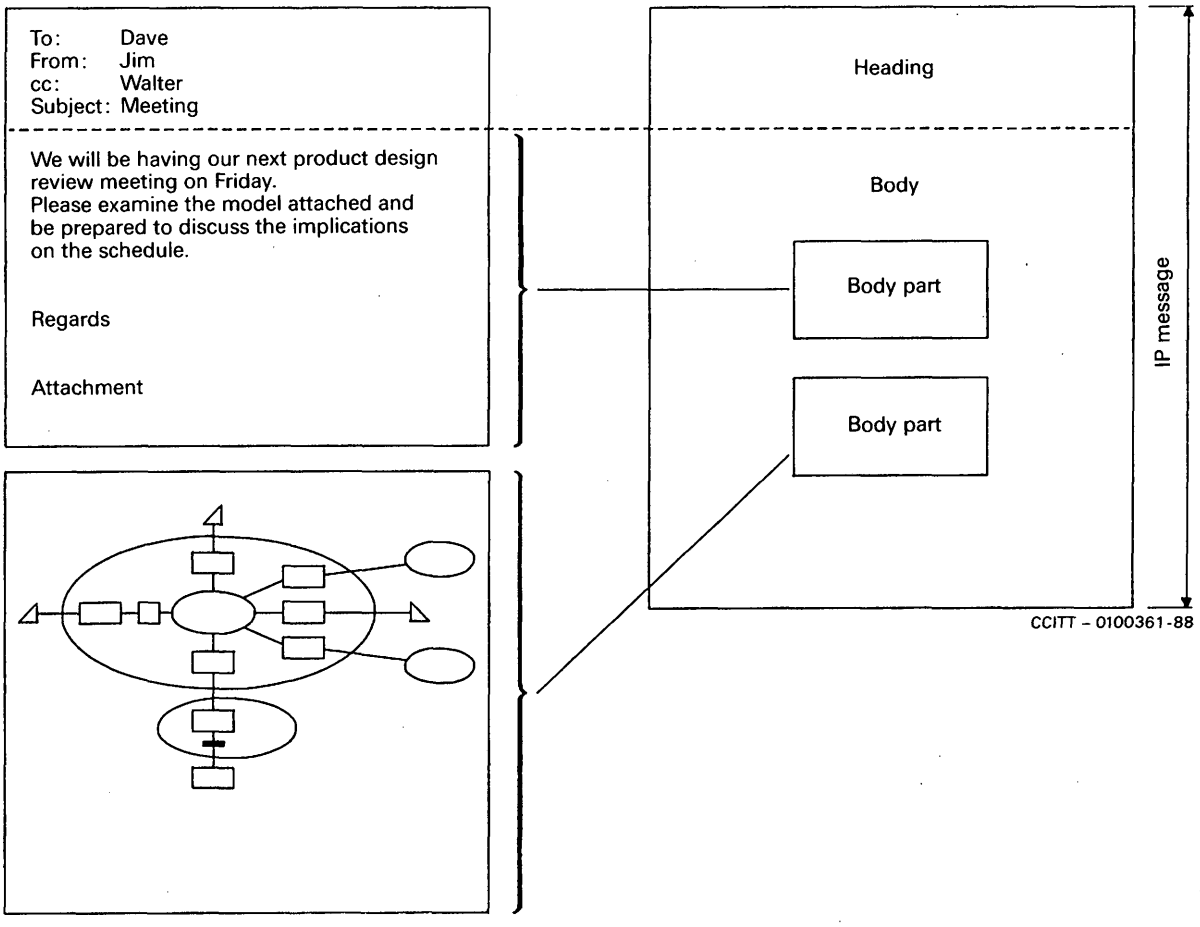


FIGURE 9/X.400

IP-message structure for a typical memo

9.3 *IP-notifications*

In the IPM service a user can request a notification of receipt or non-receipt of a message by a recipient. These notifications are requested by an originator and are generated as a result of some (such as reading/not reading the message) recipient action. In certain cases the non-receipt notification is generated automatically by the recipient's UA.

10 Intercommunication with physical delivery services

10.1 *Introduction*

The value of message handling systems can be increased by connecting them to physical delivery (PD) systems such as the traditional postal service. This will allow for the physical (e.g., hardcopy) delivery of messages originated within MHS to recipients outside of MHS, and in some cases will allow for the return of notifications from the PD service to an MHS originator. The ability for origination of messages in the PD service for submission to MHS through the PDAU is for further study. The capability of intercommunication between PD and MH services is an optional capability of MHS, and is applicable to any application such as IPM. All users of MHS will have the ability to generate messages for subsequent physical delivery. Figure 10/X.400 shows the functional model of this interworking. Provision of intercommunication between public message handling services offered by Administrations and PD services is described in Recommendation F.415. The elements of service describing the features of this intercommunication are defined in Annex B and classified in § 19.

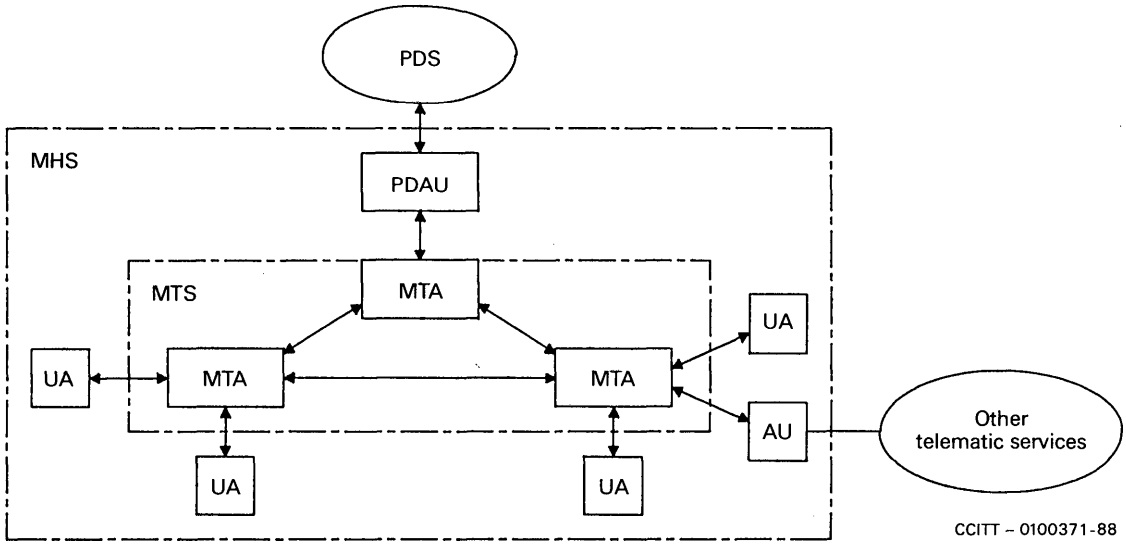


FIGURE 10/X.400

Functional model MHS-PDS interworking

A physical delivery system is a system, operated by a management domain, that transports and delivers physical messages. A physical message is a physical object comprising a relaying envelope and its content. An example of a PDS is the postal service. An example of a physical message is a paper letter and its enclosing paper envelope.

A physical delivery access unit (PDAU) converts an MH user's message to physical form, a process called physical rendition. An example of this is the printing of a message and its automatic enclosure in a paper envelope. The PDAU passes the physically rendered message to a PDS for further relaying and eventual physical delivery.

A PDAU can be viewed as a set of UAs, each UA being identified by a postal address. To perform its functions, a PDAU must support submission (notifications) and delivery interactions with the MTS, and also cooperate with other UAs. MH/PD service intercommunication is thus provided as part of the message transfer service.

To enable MH users to address messages, to be delivered physically by a PDS, an address form appropriate for this exists and is described in § 12.

10.2 *Organizational configurations*

Possible organizational mappings of the functional model described above are shown in Figure 11/X.400. In each model (A & B), the term PD domain denotes the domain of responsibility of an organization providing a PD service. In A, the PD domain comprises an MD and a PDS. The boundary between the PD domain and the rest of MHS is a boundary between MDs. In B, the PD domain comprises only the PDS; the PDAU is not part of the PD domain. The boundary between the PD domain and MHS lies at the point where the PDAU passes physical messages to the PDS.

11 Specialized access

11.1 *Introduction*

The functional model of MHS (Figure 1/X.400) contains access units (AUs) to allow access between MHS and other communication systems and services. The model shows a generic access unit between MHS and telematic services.

Also shown in a physical delivery access unit to allow for physical delivery of MHS messages to recipients without the need for terminal access to MHS. The access to physical delivery services is available to any application carried by the MTS, through a PDU described in § 10.

Other forms of access are described below.

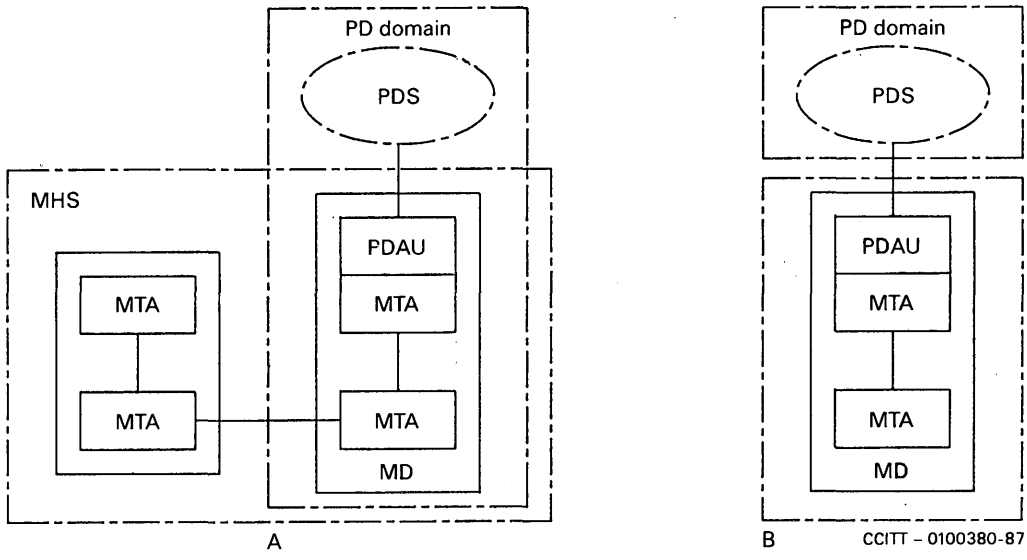


FIGURE 11/X.400

Configurations for MH/PD service intercommunication

11.2 *Teletex access*

11.2.1 *Registered access to the IPM service*

The specialized access unit defined for telematic access – telematic agent (TLMA) caters specially for teletex (TTX) terminals. This TLMA provides for teletex access to the IPM service as shown in Figure 7/X.400. The technical provisions of this access are defined in Recommendation T.330. The TLMA enables users of teletex terminals to participate fully in the IPM service.

11.2.2 *Non-registered (public) access to the IPM service*

The specialized access unit defined for telematic access – telematic agent (TLMA) also provides for public access to the IPM service for TTX users who are not registered users of the IPM service. This is shown in Figure 7/X.400. The technical provisions of this access are defined in Recommendation T.330. The intercommunication between the IPM service and the teletex service is defined in Recommendation F.422.

11.3 *Telex access*

11.3.1 *Registered access to the IPM service*

A telex access unit (TLXAU) is defined in the technical Recommendations to allow the intercommunication between IPM users and telex users. To provide a service with this type of AU is a national matter.

11.3.2 *Non-registered (public) access to the IPM service*

A specialized access unit is defined to allow the intercommunication between IPM users and telex users. This AU provides for public access to the IPM service for telex users who are not registered users of the IPM service, and is called a public telex access unit (PTLXAU). This is shown in Figure 7/X.400. The telex users are not subscribers to the IPM service, but use some of the features of the IPM service to pass messages to IPM users. IPM users can also send messages to telex users via this AU. The intercommunication between the IPM service and the telex service is defined in Recommendation F.421.

Note – Other types of access units are for further study (e.g., facsimile, videotex, etc.).

12 Naming and addressing

12.1 *Introduction*

In an MHS, the principal entity that requires naming is the user (the originator and recipient of messages). In addition, distribution lists (DLs) have names for use in MHS. Users of MHS and DLs are identified by O/R names. O/R names are comprised of directory names and/or addresses, all of which are described in this clause.

12.2 *Directory names*

Users of the MH service, and DLs, can be identified by a name, called a directory name. A directory name must be looked up in a directory to find out the corresponding O/R address. The structure and components of directory names are described in the X.500-Series of Recommendations.

A user can access a directory system directly to find out the O/R address of a user, or O/R addresses of the members of a DL (both of which are outside the scope of these Recommendations). As an alternative, a user can use the directory name and have MHS access a directory to resolve the corresponding O/R address or addresses automatically as described in § 14.

An MH user or DL will not necessarily have a directory name, unless they are registered in a directory. As directories become more prevalent, it is expected that directory names will be the preferred method of identifying MHS users to each other.

12.3 *O/R names*

Every MH user or DL will have one or more O/R name(s). An O/R name comprises a directory name, and O/R address, or both.

Either or both components of an O/R name can be used on submission of a message. If only the directory name is present, MHS will access a directory to attempt to determine the O/R address, which it will then use to route and deliver the message. If a directory name is absent, it will use the O/R address as given. When both are given on submission, MHS will use the O/R address, but will carry the directory name and present both to the recipient. If the O/R address is invalid, it will then attempt to use the directory name as above.

12.4 *O/R addresses*

An O/R address contains information that enables MHS to uniquely identify a user to deliver a message or return a notification to him. (The prefix "O/R" recognizes the fact that the user can be acting as either the originator or recipient of the message or notification in question.)

An O/R address is a collection of information called attributes. Recommendation X.402 specifies a set of standard attributes from which O/R addresses can be constructed. Standard attributes mean that their syntax and semantics are defined in Recommendation X.402. In addition to standard attributes, and to cater for existing messaging systems, there are domain defined attributes whose syntax and semantics are defined by management domains.

Various forms of O/R addresses are defined, each serving their own purpose. These forms and their purpose are as follows:

- *Mnemonic O/R address:* Provides a user-friendly means of identifying users in the absence of a directory. It is also used for identifying a distribution list.
- *Terminal O/R address:* Provides a means of identifying users with terminals belonging to various networks.
- *Numeric O/R address:* Provides a means of identifying users by means of numeric keypads.
- *Postal O/R address:* Provides a means of identifying originators and recipients of physical messages.

13 MHS use of directory

13.1 *Introduction*

The directory defined by the X.500-Series of Recommendations provides capabilities useful in the use and provision of a variety of telecommunication services. This clause describes how a directory can be used in messages handling. Details can be found in other X.400 Recommendations.

The directory capabilities used in message handling fall into the following four categories:

a) *User-friendly naming:* The originator or recipient of a message can be identified by means of his directory name, rather than his machine oriented O/R address. At any time MHS can obtain the latter from the former by consulting the directory.

b) *Distribution lists (DLs):* A group whose membership is stored in the directory can be used as a DL. The originator simply supplies the name of the list. At the DL's expansion point MHS can obtain the directory names (and then the O/R addresses) of the individual recipients by consulting the directory.

c) *Recipient UA capabilities:* MHS capabilities of a recipient (or originator) can be stored in his directory entry. At any time MHS can obtain (and then act upon) those capabilities by consulting the directory.

d) *Authentication:* Before two MHS functional entities (two MTAs, or a UA and an MTA) communicate with one another, each establishes the identity of the other. This can be done by using authentication capabilities of MHS based on information stored in the directory.

Besides the above, one user can directly access the directory, for example, to determine the O/R address or MHS capabilities of another. The recipient's directory name is supplied to the directory, which returns the requested information.

13.2 *Functional model*

Both UAs and MTAs can use the directory. A UA can present the directory with the directory name of the intended recipient, and obtain from the directory the recipient's O/R address. The UA can then supply both the directory name and the O/R address to the MTS. Another UA can supply just the recipient's directory name to the MTS. The MTS would then itself ask the directory for the recipient's O/R address and add it to the envelope. The originating MTA normally carries out the name to O/R address look up.

A functional model depicting the above is shown in Figure 12/X.400.

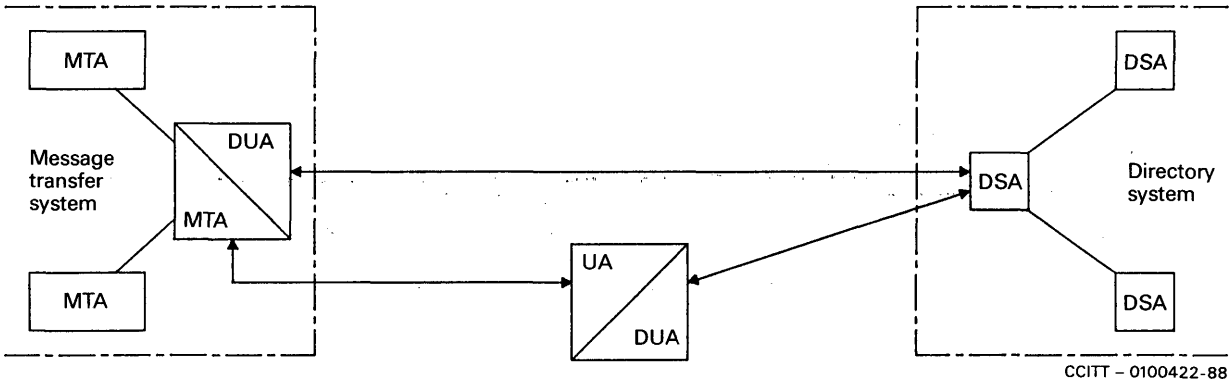


FIGURE 12/X.400

Functional model of MHS-directory interworking

13.3 *Physical configurations*

Some possible physical configurations of the above functional model are shown in Figure 13/X.400. Where a directory user agent (DUA) and directory system agent (DSA) reside in physically separate systems, a standard directory protocol, defined in the X.500-Series of Recommendations, governs their interactions. It will often be desirable to physically co-locate a UA or MTA with a DUA/DSA. However, other physical configurations are also possible.

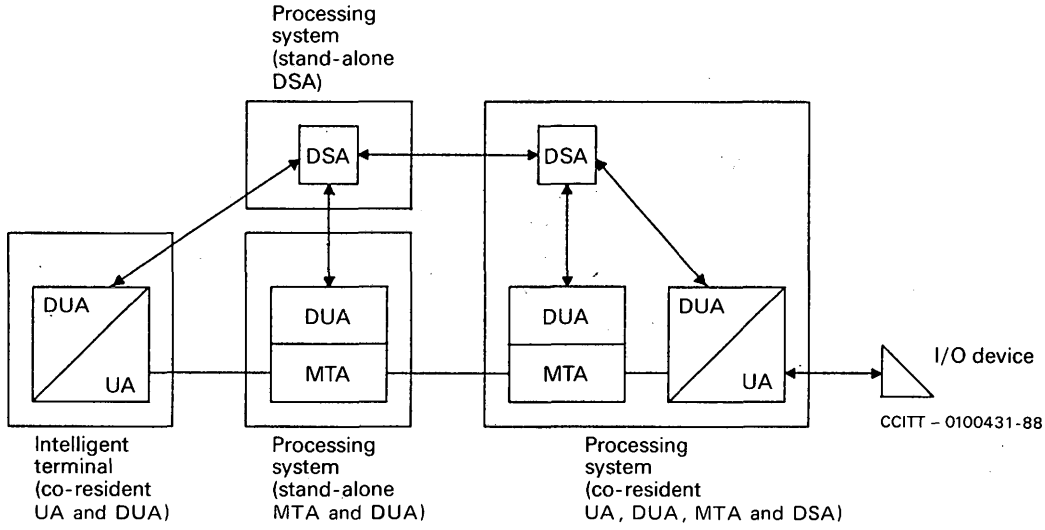


FIGURE 13/X.400

Physical configurations for MHS-directory interworking

14 Distribution lists in MHS

14.1 *Introduction*

The ability to make use of a distribution list (DL) is an optional capability of MHS provided through the MT service. DL expansion allows a sender to have a message transmitted to a group of recipients, by naming the group instead of having to enumerate each of the final recipients.

14.2 *Properties of a DL*

The properties of a DL can be described as follows:

- *DL members:* Users and other DLs that will receive messages addressed to the DL.
- *DL submit permission:* A list of users and other DLs which are allowed to make use of the DL to send messages to the DL's members.
- *DL expansion point:* Each DL has an unambiguous O/R address. This O/R address identifies the expansion point, which is the domain or MTA where the names of the members of the DL are added to the recipient list. The message is transported to the expansion point before expansion as shown in Figure 14/X.400.
- *DL owner:* A user who is responsible for the management of a DL.

14.3 *Submission*

Submission of a message to a DL is similar to the submission of a message to a user. The originator can include in the DL's O/R name, the directory name, the O/R address, or both (see § 12 for details). The originator need not be aware that the O/R name used is that of a DL. The originator can, however, through use of the element of service, DL expansion prohibited, prohibit the MTS from expanding a message unknowingly addressed to a DL.

14.4 *DL use of a directory*

A directory may or may not be used to store information about the properties of a DL. Among the information that can be stored are the following: DL members, DL owner, DL submit permission and the DL expansion point.

14.5 *DL expansion*

At the expansion point, the MTA responsible for expanding the DL will:

a) Look up the information about the DL, e.g. in the directory, using access rights granted to the MTA. (*Note* – Since this is done by the MTA at the expansion point, support of DLs in MHS does not require a globally interconnected directory).

b) Verify whether expansion is allowed by checking the identity of the sender against the DL's submit permission.

c) If expansion is allowed, add the members of the DL to the list of recipients of the message and transmit the message to them.

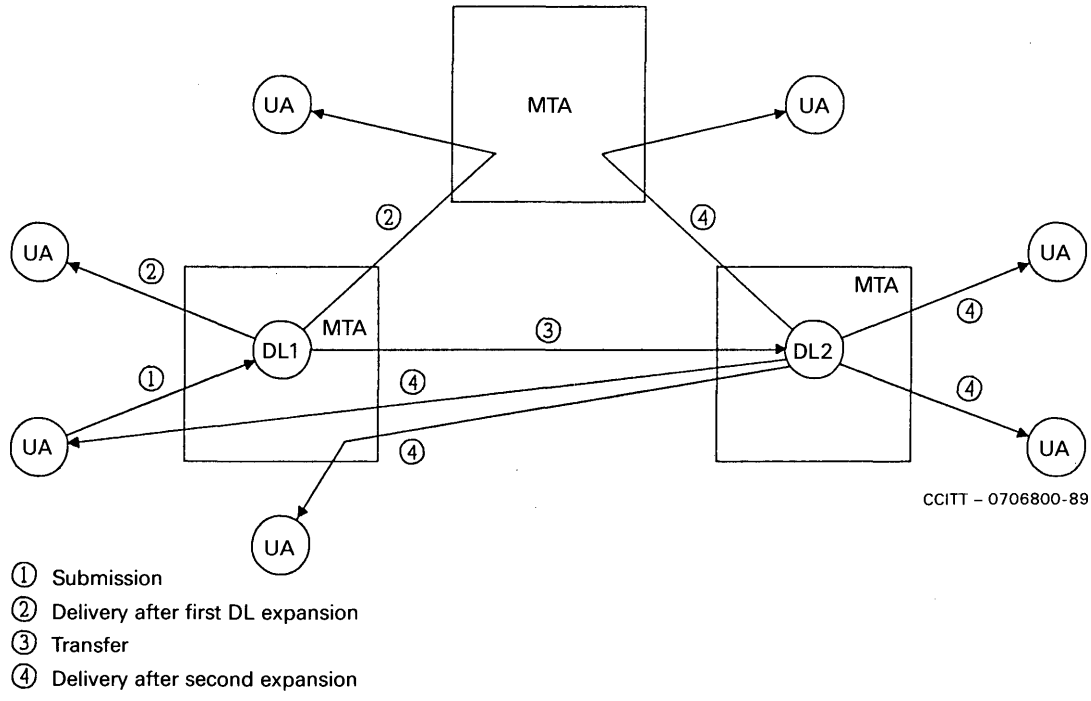


FIGURE 14/X.400

Distribution list expansion

14.6 *Nesting*

A member of a DL can be another DL as shown in Figure 14/X.400. In this case the message is forwarded from the expansion point of the parent DL to the expansion point of the member DL for further expansion. Thus during each expansion, only the members of a single DL are added to the message.

During expansion of a nested DL, the identity of the parent DL (e.g., DL1 in Figure 14/X.400) rather than that of the message originator, is compared against the submit permission of the member DL (e.g., DL2 in Figure 14/X.400).

Note – DL structures can be defined which reference a particular nested DL more than once at different levels of the nesting. Submission to such a parent DL can cause a recipient to receive multiple copies of the same message. The same result can occur if a message is addressed to multiple DLs which contain a common member. Correlation of such copies can be done at the recipient's UA, and/or in the MS.

14.7 *Recursion control*

If a certain DL is directly or indirectly a member of itself (a situation which can validly arise), or when DLs are combined with redirection, then a message might get back to the same list and potentially circulate infinitely. This is detected by the MTS and prevented from occurring.

14.8 *Delivery*

On delivery of the message, the recipient will find out that he received the message as a member of a DL, and through which DL, or chain or DLs he got the message.

14.9 *Routing loop control*

A message can be originated in one domain/MTA, expanded in a second domain/MTA, and then sent back to a DL member in the first domain/MTA. The MTS will not treat this as a routing loop error.

14.10 *Notifications*

Delivery and non-delivery notifications can be generated both at the DL expansion point (e.g. if submit permission is denied), and at delivery to the ultimate recipient.

When a message coming from a DL generates a notification, this notification is sent to the DL from which the message came. The DL will then, depending on the policy of the list, forward the notification to the owner of the list, to the DL or originator from which it got the message, or both, as shown in Figure 15/X.400.

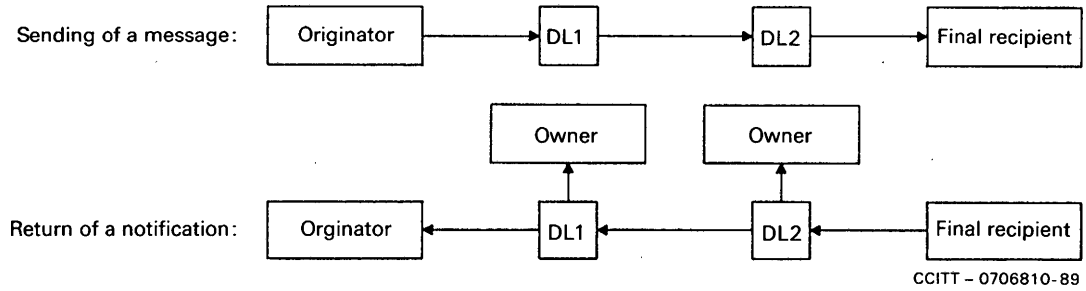


FIGURE 15/X.400

DL notifications

Note – When notifications are sent to the originator after DL expansion, the originator can receive many delivery/non-delivery notifications for one originator specified recipient (the DL itself). The originator can even receive more than one notification from an ultimate recipient, if that recipient received the message more than once via different lists.

14.11 *DL handling policy*

An MTA may or may not provide different policies on DL handling. Such policies will control whether notifications generated at delivery to DL members should be propagated back through the previous DL, or to the originator if no such previous DL, and/or to this list owner. If the policy is such that notifications are to be sent only to the list owner, then the originator will receive notifications if requested, only during expansion of that DL. In order to accomplish this restriction, the MTS will, while performing the expansion, reset the notification requests according to the policy for the list.

15 Security capabilities of MHS

15.1 *Introduction*

The distributed nature of MHS makes it desirable that mechanisms are available to protect against various security threats that can arise. The nature of these threats and the capabilities to counter them are highlighted below.

15.2 *MHS security threats*

15.2.1 *Access threats*

Invalid user access into MHS is one of the prime security threats to the system. If invalid users can be prevented from using the system, then the subsequent security threat to the system is greatly reduced.

15.2.2 *Inter-message threats*

Inter-message threats arise from unauthorized agents who are external to the message communication, and can manifest themselves in the following ways;

- *Masquerade:* A user who does not have proof of whom he is talking to can be easily misled by an imposer into revealing sensitive information.
- *Message modification:* A genuine message which has been modified by an unauthorized agent while it was transferred through the system can mislead the message recipient.
- *Replay:* Messages whose originators and contents are genuine can be monitored by an unauthorized agent and could be recorded to be replayed to the message's intended recipient at a later date. This could be done in order to either extract more information from the intended recipient or to confuse him.
- *Traffic analysis:* Analysis of message traffic between MH users can reveal to an eavesdropper how much data (if any) is being sent between users and how often. Even if the eavesdropper cannot determine the actual contents of the messages, he can still deduce a certain amount of information from the rate of traffic flow (e.g. continuous, burst, sporadic or none).

15.2.3 *Intra-message threats*

Intra-message threats are those performed by the actual message communication participants themselves, and can manifest themselves in the following ways:

- *Repudiation of messages:* One of the actual communication participants can deny involvement in the communication. This could have serious implications if financial transactions were being performed via MHS.
- *Security level violation:* If a management domain within MHS employs different security clearance levels (e.g. public, personal, private and company confidential) then users must be prevented from sending or receiving any messages for which they have an inadequate security clearance level if the management domain's security is not to be compromised.

15.2.4 *Data store threats*

An MHS has a number of data stores within it that must be protected from the following threats:

- *Modification of routing information:* Unauthorized modification of the directory's contents could lead to messages being mis-routed or even lost while unauthorized modification to the deferred delivery data store or the hold for delivery data store could mislead or confuse the intended recipient.
- *Preplay:* An unauthorized agent could make a copy of a deferred delivery message and send this copy to the intended recipient while the original was still being held for delivery in the MTA. This could fool the message recipient into replying to the message originator before the originator was expecting a reply or simply mislead or confuse the original intended message recipient.

15.3 *Security model*

Security features can be provided by extending the capabilities of the components in the message handling system to include various security mechanisms.

There are two aspects to security in message handling: secure access management and administration, and secure messaging.

15.3.1 *Secure access management and administration*

The capabilities in this section cover the establishment of an authenticated association between adjacent components, and the setting up of security parameters for the association. This can be applied to any pair of components in the message handling system: UA/MTA, MTA/MTA, MS/MTA, etc.

15.3.2 *Secure messaging*

The capabilities in this section cover the application of security features to protect messages in the message handling system in accordance with a defined security policy. This includes elements of service enabling various components to verify the origin of messages and the integrity of their content, and elements of service to prevent unauthorized disclosure of the message content.

The capabilities in this section cover the application of security features to protect messages directly submitted to the message transfer system by a user agent, message store, or an access unit. They do not cover the application of security features to protect communication between users and the message handling system, or MH user-to-MH user communication (a large part of MH user-to-MH user communication is protected between two UAs). Thus they do not apply, for example, to communication between a remote user's terminal and its UA, or to communication between these users' terminal equipment and other users in the MHS. Security capabilities to protect MH user-to-MH user communication are for further study.

Many of the secure messaging elements of service provide an originator to recipient capability, and require the use of user agents with security capabilities. They do not require the use of a message transfer system with security features. (As an example, content confidentiality can be applied by enciphering the message content by the originator, and deciphering by the recipient, with various security parameters transferred within the message envelope. Such a message can be transferred by an MTS which can handle the format of the content (unformatted octets), and transparently handle the security fields in the envelope.)

Some of the secure messaging elements of service involve an interaction with the message transfer system, and require the use of message transfer agents with security capabilities. (As an example, non-repudiation of submission requires the MTA, to which the message is submitted, to contain mechanisms to generate a proof of submission field.)

Some of the secure messaging elements of service apply to the MS as well as UAs and MTAs, such as message security labelling. In general, however, the MS is transparent to security features that apply between the originators' and the recipients' UAs.

The scope of the secure messaging elements of service is given in Table 2/X.400. This describes the elements of service in terms of which MHS component is the "provider" or which is the "user" of the security service. For example, probe origin authentication is provided by the originating UA, and can be used by the MTAs through which the probe passes.

This Recommendation describes the use of security services by the UA, and the MTA. How these features are applied to access units is for further study.

15.4 *MHS security capabilities*

The elements of service describing the security features of MHS are defined in Annex B, and classified in § 19. An overview of these capabilities is as follows:

- *Message origin authentication:* Enables the recipient, or any MTA through which the message passes, to authenticate the identity of the originator of a message.
- *Report origin authentication:* Allows the originator to authenticate the origin of a delivery/non-delivery report.
- *Probe origin authentication:* Enables any MTA through which the probe passes, to authenticate the origin of the probe.
- *Proof of delivery:* Enables the originator of a message to authenticate the delivered message and its content, and the identity of the recipient(s).
- *Proof of submission:* Enables the originator of a message to authenticate that the message was submitted to the MTS for delivery to the originally specified recipient(s).
- *Secure access management:* Provides for authentication between adjacent components, and the setting up of the security context.
- *Content integrity:* Enables the recipient to verify that the original content of a message has not been modified.
- *Content confidentiality:* Prevents the unauthorized disclosure of the content of a message to a party other than the intended recipient.

- *Message flow confidentiality:* Allows the originator of a message to conceal the message flow through MHS.
- *Message sequence integrity:* Allows the originator to provide to a recipient proof that the sequence of messages has been preserved.
- *Non-repudiation of origin:* Provides the recipient(s) of a message with proof of origin of the message and its content which will protect against any attempt by the originator to falsely deny sending the message or its content.
- *Non-repudiation of delivery:* Provides the originator of a message with proof of delivery of the message which will protect against any attempt by the recipient(s) to falsely deny receiving the message of its content.
- *Non-repudiation of submission:* Provides the originator of a message with proof of submission of the message, which will protect against any attempt by the MTS to falsely deny that the message was submitted for delivery to the originally specified recipient(s).
- *Message security labelling:* Provides a capability to categorize a message, indicating its sensitivity, which determines the handling of a message in line with the security policy in force.

TABLE 2/X.400

Provision and use of secure messaging elements of service by MHS components

Elements of service	Originating MTS user	MTS	Recipient MTS user
Message origin authentication	P	U	U
Report origin authentication	U	P	–
Probe origin authentication	P	U	–
Proof of delivery	U	–	P
Proof of submission	U	P	–
Secure access management	P	U	P
Content integrity	P	–	U
Content confidentiality	P	–	U
Message flow confidentiality	P	–	–
Message sequence integrity	P	–	U
Non-repudiation of origin	P	–	U
Non-repudiation of submission	U	P	–
Non-repudiation of delivery	U	–	P
Message security labelling	P	U	U

P The MHS component is a provider of the service.

U The MHS component is a user of the service.

15.5 *Security management*

Aspects of an asymmetric key management scheme to support the above features are provided by the directory system authentication framework, described in Recommendation X.509. The directory stores certified copies of public keys for MHS users which can be used to provide authentication and to facilitate key exchange for use in data confidentiality and data integrity mechanisms. The certificates can be read from the directory using the directory access protocol described in Recommendation X.519.

Recommendations for other types of key management schemes, including symmetric encryption, to support the security features are for further study.

16 Conversion in MHS

The MTS provides conversion functions to allow users to input messages in one or more encoded formats, called encoded information types (EITs), and have them delivered in other EITs to cater to users with various UA capabilities and terminal types. This capability is inherent in the MTS and increases the possibility of delivery by tailoring the message to the recipient's terminal capabilities. The EITs standardized in MHS are listed in Recommendation X.411. Conversions and the use of the elements of service relating to conversion are available for EITs not defined in Recommendation X.411, but supported by certain domains, either bilaterally between these domains or within a domain itself.

MHS users have some control over the conversion process through various elements of service as described in Annex B. These include the ability for a user to explicitly request the conversion required or as a default to let the MTS determine the need for conversion, and the type of conversion performed. Users also have the ability to request that conversion not be performed or that conversion not be performed if loss of information will result. When the MTS performs conversion on a message it informs the UA to whom the message is delivered that conversion took place and what the original EITs were.

The conversion process for IP-messages can be performed on body parts of specific types if they are present in a message. The general aspects of conversion and the specific conversion rules for conversion between different EITs are detailed in Recommendation X.408.

Recommendation X.408 deals with conversion for the following: telex, IA5 text, teletex, G3fax, G4 Class1, videotex, voice, and mixed mode.

17 Use of the MHS in provision of public services

The message handling system is used in the provision of public MH services that are offered by Administrations for use by their subscribers. These public MH services are defined in the F.400-Series Recommendations and include:

- the public message transfer service (Rec. F.410);
- the public interpersonal messaging service (Rec. F.420).

In addition complementary public services are offered by Administrations to allow for the intercommunication between CCITT services and the public MH services mentioned above, as follows:

- intercommunication with public physical delivery services (Rec. F.415).
- intercommunication between the IPM service and the telex service (Rec. F.421);
- intercommunication between the IPM service and the teletex service (Rec. F.422);

Recommendation F.401 describes the naming and addressing aspects for public MH services.

18 Purpose

Elements of service are particular features, functions, or capabilities of MHS. All the elements of service applicable for MHS are defined in Annex B, where they are listed in alphabetical order with a corresponding reference number. The realization of these elements of service in MHS are described in other Recommendations in the X.400 Series.

Elements of service are associated with the various services provided in MHS. There are elements of service for the message transfer service which provide for a basic capability for sending and receiving messages between UAs. There are elements of service for the interpersonal messaging service which provide for the sending and receiving of messages between a particular class of UAs called IPM UAs. There are elements of service for the physical delivery service, enabling MH users to send messages and have them delivered in a physical medium to non-MH users. There are elements of service specifically available for the use of message stores.

The elements of service for the IPM service include those available for the MT service, the PD service, and the message store as well as specific ones applicable to the IPM service.

Table 3/X.400 lists all the elements of service available in MHS, shows what service they are specifically associated with of the presently defined services, MT service, IPM service, and PD service, or whether they are specific to the message store, and gives the corresponding reference number to the definition in Annex B.

TABLE 3/X.400

MHS elements of service

Elements of service	MT	IPM	PD	MS	Annex B reference
Access management	X				B.1
Additional physical rendition			X		B.2
Alternate recipient allowed	X				B.3
Alternate recipient assignment	X				B.4
Authorizing users indication		X			B.5
Auto-forwarded indication		X			B.6
Basic physical rendition			X		B.7
Blind copy recipient indication		X			B.8
Body part encryption indication		X			B.9
Content confidentiality	X				B.10
Content integrity	X				B.11
Content type indication	X				B.12
Conversion prohibition	X				B.13
Conversion prohibition in case of loss information	X				B.14
Converted indication	X				B.15
Counter collection			X		B.16
Counter collection with advice			X		B.17
Cross-referencing indication		X			B.18
Deferred delivery	X				B.19
Deferred delivery cancellation	X				B.20
Delivery notification	X				B.21
Delivery time stamp indication	X				B.22
Delivery via Bureaufax service			X		B.23
Designation of recipient by directory name	X				B.24
Disclosure of other recipients	X				B.25
DL expansion history indication	X				B.26
DL expansion prohibited	X				B.27
EMS (express mail service)			X		B.28
Expiry date indication		X			B.29
Explicit conversion	X				B.30
Fowarded IP-message indication		X			B.31
Garde of delivery selection	X				B.32
Hold for delivery	X				B.33
Implicit conversion	X				B.34
Importance indication		X			B.35
Incomplete copy indication		X			B.36
IP-message identification		X			B.37
Language indication		X			B.38
Latest delivery designation	X				B.39
Message flow confidentiality	X				B.40
Message identification	X				B.41
Message origin authentication	X				B.42
Message security labelling	X				B.43
Message sequence integrity	X				B.44
Multi-destination delivery	X				B.45
Multi-part body		X			B.46
Non-delivery notification	X				B.47

TABLE 3/X.400 *(cont.)*

Elements of service	MT	IPM	PD	MS	Annex B reference
Non-receipt notification request indication		X			B.48
Non-repudiation of delivery	X				B.49
Non-repudiation of origin	X				B.50
Non-repudiation of submission	X				B.51
Obsoleting indication		X			B.52
Ordinary mail			X		B.53
Original encoded information types indication	X				B.54
Originator indication		X			B.55
Originator requested alternate recipient	X				B.56
Physical delivery notification by MHS			X		B.57
Physical delivery notification by PDS			X		B.58
Physical forwarding allowed			X		B.59
Physical fowarding prohibited			X		B.60
Prevention of non-delivery notification	X				B.61
Primary and copy recipients indication		X			B.62
Probe	X				B.63
Probe origin authentication	X				B.64
Proof of delivery	X				B.65
Proof of submission	X				B.66
Receipt notification request indication		X			B.67
Redirection disallowed by originator	X				B.68
Redirection of incoming messages	X				B.69
Registered mail			X		B.70
Registered mail to addressee in person			X		B.71
Reply request indication		X			B.72
Replying IP-message indication		X			B.73
Report origin authentication	X				B.74
Request for forwarding address			X		B.75
Requested delivery method	X				B.76
Restricted delivery	X				B.77
Return of content	X				B.78
Secure access management	X				B.79
Sensitivity indication		X			B.80
Special delivery			X		B.81
Stored message alert				X	B.82
Stored message auto-forward				X	B.83
Stored message deletion				X	B.84
Stored message fetching				X	B.85
Stored message listing				X	B.86
Stored message summary				X	B.87
Subject indication		X			B.88
Submission time stamp indication	X				B.89
Type body		X			B.90
Undeliverable mail with return of physical message			X		B.91
Use of distribution list	X				B.92
User/UA capabilities registration	X				B.93

19 Classification

19.1 *Purpose of classification*

The elements of service of MHS are classified either as belonging to a basic (also called base for PD and MS) service, or as optional user facilities. Elements of service belonging to a basic service are inherently part of that service – they constitute the basic service and are always provided and available for use of that service.

Other elements of service, called optional user facilities, can be selected by the subscriber or user, either on a per-message basis, or for an agreed contractual period of time. Each optional user facility is classified as either essential or additional. Essential (E) optional user facilities are to be made available to all MH users. Additional (A) optional user facilities can be made available for national use, and for international use on the basis of bilateral agreement.

19.2 *Basic message transfer service*

The basic MT service enables a UA to submit and to have messages delivered to it. If a message cannot be delivered, the originating UA is so informed through a non-delivery notification. Each message is uniquely and unambiguously identified. To facilitate meaningful communication, a UA can specify the encoded information type(s) that can be contained in messages which are delivered to it. The content type and original encoded information type(s) of a message and an indication of any conversions that have been performed, and the resulting encoded information type(s), are supplied with each delivered message. In addition, the submission time and delivery time are supplied with each message. The MT elements of service belonging to the basic MT service are listed in Table 4/X.400.

TABLE 4/X.400

Elements of service belonging to the basic MT service

Elements of service	Annex B ref.
Access management	B.1
Content type indication	B.12
Converted indication	B.15
Delivery time stamp indication	B.22
Message indication	B.41
Non-delivery notification	B.47
Original encoded information types indication	B.54
Submission time stamp indication	B.89
User/UA capabilities registration	B.93

19.3 *MT service optional user facilities*

Optional user facilities for the MT service can be selected on a per-message basis, or for an agreed period of time. Each optional user facility is classified as either essential or additional as described in § 19.1. Table 5/X.400 lists the elements of service comprising the optional user facilities of the MT service with their classification and their availability (PM: per-message; CA: contractual agreement). Optional user facilities for the PD service and the message store, while forming a part of the MT service optional user facilities, are not listed in this table because they are subject to either a PDAU or an MS being supplied, and are given separate classifications in Tables 6/X.400-9/X.400.

TABLE 5/X.400

MT service optional user facilities

Elements of service	Classification	Available	Annex B ref.
Alternate recipient allowed	E	PM	B.3
Alternate recipient assignment	A	CA	B.4
Content confidentiality	A	PM	B.10
Content integrity	A	PM	B.11
Conversion prohibition	E	PM	B.13
Conversion prohibition in case of loss of information	A	PM	B.14
Deffered delivery	E	PM	B.19
Defered delivery cancellation	E	PM	B.20
Delivery notification	E	PM	B.21
Designation of recipient by directory name	A	PM	B.24
Disclosure of other recipients	E	PM	B.25
DL expansion history indication	E	PM	B.26
DL expansion prohibited	A	PM	B.27
Explicit conversion	A	PM	B.30
Grade of delivery selection	E	PM	B.32
Hold for delivery	A	CA	B.33
Implicit conversion	A	CA	B.34
Lastest delivery designation	A	PM	B.39
Message flow confidentiality	A	PM	B.40
Message origin authentication	A	PM	B.42
Message security labelling	A	PM	B.43
Message sequence integrity	A	PM	B.44
Multi-destination delivery	A	PM	B.45
Non-repudiation of delivery	A	PM	B.49
Non-repudiatation of origin	A	PM	B.50
Non-repudiation of submission	A	PM	B.51
Originator requested alternate recipient	A	PM	B.56
Prevention of non-delivery notification	A	PM	B.61
Probe	E	PM	B.63
Probe origin authentication	A	PM	B.64
Proof of delivery	A	PM	B.65
Proof of submission	A	PM	B.66
Redirection disallowed by originator	A	PM	B.68
Redirection of incoming messages	A	CA	B.69
Report origin authentication	A	PM	B.74
Requested delivery method	E [a)]	PM	B.76
Restricted delivery	A	CA	B.77
Return of content	A	PM	B.78
Secure access management	A	CA	B.79
Use of distribution list	A	PM	B.92

[a)] Does not imply the provision of all delivery methods which may be requested.

19.4 *Base MH/PD service intercommunication*

The base MH/PD service intercommunication can be supplied, to enhance the MT service, and enables messages to be delivered to recipients in a physical (typically hard copy) format via a physical delivery service such as the postal service. This capability is applicable for use by any application making use of the MT service. The MH/PD elements of service belonging to the base MH/PD service intercommunication are available on a per-recipient basis and are listed in Table 6/X.400. When this intercommunication is provided, through a PDAU, all the elements of service shown in Table 6/X.400 shall be supported.

TABLE 6/X.400

Elements of service belonging to the base MH/PD service intercommunication

Elements of service	Annex B ref.
Basic physical rendition	B.7
Ordinary mail	B.53
Physical forwarding allowed	B.59
Undeliverable mail with return of physical message	B.91

19.5 *Optional user facilities for MH/PD service intercommunication*

Base MH/PD elements of servie § 19.4) together with the optional user facilities listed below, can be used together for the provision of the MH/PD service intercommunication. This capability is applicable for use by any application making use of the enhanced MT service. These optional user facilities can be selected on a per-recipient basis and are listed in Table 7/X.400.

TABLE 7/X.400

Optional user facilities for MH/PD service intercommunication

Elements of service	Classification	Annex B ref.
Additional physical rendition	A	B.2
Counter collection	E	B.16
Counter collection with advice	A	B.17
Delivery via Bureaufax service	A	B.23
EMS (express mail service) [a]	E	B.28
Physical delivery notification by MHS	A	B.57
Physical delivery notification by PDS	A	B.58
Notification forwarding prohibited	A	B.60
Registered mail	A	B.70
Registered mail to addressee in person	A	B.71
Request for forwarding address	A	B.75
Special delivery [a]	E	B.81

[a] At least one or the other shall be supported by the PDAU and the associated PDS.

19.6 *Base message store*

The base message store is optionally available to provide for storage and management of incoming messages acting as an intermediary between a UA and an MTA. The MS is applicable for use in any application making use of the MT service. The elements of service belonging to the base message store are listed in Table 8/X.400. When an MS is provided, all the elements of service shown in Table 8/X.400 shall be supported.

TABLE 8/X.400

Base message store

Elements of service	Annex B ref.
Stored message deletion	B.84
Stored message fetching	B.85
Stored message listing	B.86
Stored message summary	B.87

19.7 *MS optional user facilities*

Base MS elements of service (§ 19.6) together with the optional user facilities listed below can be used together for enhanced use of a message store. The enhanced MS is applicable for use in any application making use of the MT service. The elements of service comprising the MS optional user facilities are listed in Table 9/X.400.

TABLE 9/X.400

MS optional user facilities

Elements of service	Classification	Annex B ref.
Stored message alert	A	B.82
Stored message auto-forward	A	B.83

19.8 *Basic interpersonal messaging service*

The basic IPM service, which makes use of the MT service, enables a user to send and receive IP-messages. A user prepares IP-messages with the assistance of his user agent (UA). User agents cooperate with each other to facilitate communication between their respective users. To send an IP-message, the originating user submits the message to his UA specifying the O/R name of the recipient who is to receive the IP-message. The IP-message, which has an identifier conveyed with it, is then sent by the originator's UA to the recipient's UA via the message transfer service.

Following a successful delivery to the recipient's UA, the IP-message can be received by the recipient. To facilitate meaningful communication, a recipient can specify the encoded information type(s) contained in IP-messages that he will allow to be delivered to his UA. The original encoded information type(s) and an indication of any conversions that have been performed and the resulting encoded information type(s) are supplied with each delivered IP-message. In addition, the submission time and delivery time are supplied with each IP-message. Non-delivery notification is provided with the basic service. The IPM elements of service belonging to the basic IPM service are listed in Table 10/X.400.

TABLE 10/X.400

Elements of service belonging to the basic IPM service

Elements of service	Annex B ref.
Access management	B.1
Content type indication	B.12
Converted indication	B.15
Delivery time stamp indication	B.22
IP-message identification	B.37
Message identification	B.41
Non-delivery notification	B.47
Original encoded information types indication	B.54
Submission time stamp indication	B.89
Typed body	B.90
User/UA capabilities registration	B.93

19.9 *IPM service optional user facilities*

A set of the elements of service of the IPM service are optional user facilities. The optional user facilities of the IPM service, which can be selected on a per-message basis or for an agreed contractual period of time, are listed in Table 11/X.400 and Table 12/X.400, respectively. Local user facilities can be usefully provided in conjunction with some of these user facilities.

The optional user facilities of the IPM service that are selected on a per-message basis are classified for both origination and reception by UAs. If an MD offers these optional user facilities for origination by UAs, then a user is able to create and send IP-messages according to the procedures defined for the associated element of service. If an MD offers these optional user facilities for reception by UAs, MSs and AUs, then the receiving UA, MS and PDAU will be able to receive and recognize the indication associated with the corresponding element of service and to inform the user of the requested optional user facility. Each optional user facility is classified as additional (A) or essential (E) for UAs from these two perspectives.

Note – With the access protocol described in Recommendation T.330, teletex terminals are able to make use of the basic IPM service as well as of the optional user facilities provided by the message handling system.

TABLE 11/X.400

IPM optional user facilities selectable on a per-message basis

Elements of service	Origination	Reception	Annex B ref.
Additional physical rendition	A	A	B.2
Alternate recipient allowed	A	A	B.3
Authorizing users indication	A	E	B.5
Auto-forwarded indication	A	E	B.6
Basic physical rendition	A	E*	B.7
Blind copy recipient indication	A	E	B.8
Body part encryption indication	A	E	B.9
Content confidentiality	A	A	B.10
Content integrity	A	A	B.11
Conversion prohibition	E	E	B.13
Conversion prohibition in case of loss of information		N/A	B.14
Counter collection	A	E*	B.16
Counter collection with advice	A	A	B.17
Cross-referencing indication	A	E	B.18
Deffered delivery	E	N/A	B.19
Deffered delivery cancellation	A	N/A	B.20
Delivery notification	E	N/A	B.21
Delivery via Bureaufax service	A	A	B.23
Designation of recipient by directory name	A	N/A	B.24
Disclosure of other recipients	A	E	B.25
DL expansion history indication	N/A	E	B.26
DL expansion prohibited	A	A	B.27
EMS (express mail service) [a]	A	E*	B.28
Expiry date indication	A	E	B.29
Explicit conversion	A	N/A	B.30
Forwarded IP-message indication	A	E	B.31
Grade of delivery selection	E	E	B.32
Importance indication	A	E	B.35
Incomplete copy indication	A	A	B.36
Language indication	A	E	B.38
Latest delivery designation	A	N/A	B.39
Message flow confidentiality	A	N/A	B.40
Message origin authentication	A	A	B.42
Message security labelling	A	A	B.43
Message sequence integrity	A	A	B.44
Multi-destination delivery	E	N/A	B.45
Multi-part body	A	E	B.46
Non-receipt notification request indication	A	E	B.48
Non-repudiation of delivery	A	A	B.49
Non-repudiation of origin	A	A	B.50
Non-repudiation of submission	A	A	B.51
Obsoleting indication	A	E	B.52
Ordinary mail	A	E*	B.53
Originator indication	E	E	B.55
Originator requested alternate recipient	A	N/A	B.56
Physical delivery notification by MHS	A	A	B.57
Physical delivery notification by PDS	A	E*	B.58
Physical forwarding allowed	A	E*	B.59

TABLE 11/X.400 *(cont.)*

Elements of service	Origination	Reception	Annex B ref.
Physical forwarding prohibited	A	E*	B.60
Prevention of non-delivery notification	A	N/A	B.61
Primary and copy recipients indication	E	E	B.62
Probe	A	N/A	B.63
Probe origin authentication	A	A	B.64
Proof of delivery	A	A	B.65
Proof of submission	A	A	B.66
Receipt notification request indication	A	A	B.67
Redirection disallowed by originator	A	N/A	B.68
Registred mail	A	A	B.70
Registred mail to addressee in person	A	A	B.71
Reply request indication	A	E	B.72
Reply IP-message indication	E	E	B.73
Report origin authentication	A	A	B.74
Request for forwarding address	A	A	B.75
Requested delivery method	E	N/A	B.76
Return of content	A	N/A	B.78
Sensitivity indication	A	E	B.80
Special delivery [a)]	A	E*	B.81
Stored message deletion	N/A	E**	B.84
Stored message fetching	N/A	E**	B.85
Stored message listing	N/A	E**	B.86
Stored message summary	N/A	E**	B.87
Subject indication	E	E	B.88
Undeliverable mail with return of physical message	A	E*	B.91
Use of distribution list	A	A	B.92

E Essential optional user facility has to be provided.

E* Essential optional user facility only applying to PDAUs.

E** Essential optional user facility only applying to MSs.

A Additional optional user facility can be provided.

N/A Not applicable.

[a)] At least EMS or special delivery shall be supported by the PDAU and associated PDS.

Note – Bilateral agreement may be necessary in cases of reception by UA of elements of service classified by A.

TABLE 12/X.400

IPM optional user facilities agreed for a contractual period of time

Elements of service	Classification	Annex B ref.
Alternate recipient assignment	A	B.4
Hold for delivery	A	B.33
Implicit conversion	A	B.34
Redirection of incoming messages	A	B.69
Restricted delivery	A	B.77
Secure access management	A	B.79
Stored message alert	A	B.82
Stored message auto-forward	A	B.83

ANNEX A

(to Recommendation X.400)

Glossary of terms

Note – The explanations given are not necessarily definitions in the strict sense. See also the definitions in Annex B and those provided in the other X.400-Series Recommendations (especially X.402), where many entries are found. The terms have, depending on the source, varying levels of abstraction.

A.1 **access unit (AU)**

F: unité d'accès (UA)

S: unidad de acceso (AU)

In the context of a message handling system the functional object, a component of MHS, that links another communication system (e.g., a physical delivery system or the telex network) to the MTS and via which its patrons engage in message handling as indirect users.

In the context of message handling services the unit which enables users of one service to intercommunicate with message handling services, such as the IPM Service.

A.2 **actual recipient**

F: destinataire effectif

S: destinatario real

In the context of message handling a potential recipient for which delivery or affirmation takes place.

A.3 **administration**

F: administration

S: administración

In the context of CCITT an Administration (member of ITU) or a recognized private operating agency.

A.4 **administration domain name**

F: nom d'un domaine d'administration

S: nombre de dominio de administración

In the context of message handling, a standard attribute of a name form that identifies an ADMD relative to the country denoted by a country name.

A.5 **administration management domain (ADMD)**

F: domaine de gestion d'administration (DGAD)

S: dominio de gestión de administración (DGAD)

A management domain that comprises messaging systems managed by an Administration.

A.6 **alternate recipient**

F: destinataire suppléant

S: destinatario alternativo

In the context of message handling a user or distribution list to which the originator can (but need not) request that a message or probe be conveyed if and only if it cannot be conveyed to a particular preferred recipient.

A.7 **attribute**

F: attribut

S: atributo

In the context of message handling, an information item, a component of an attribute list, that describes a user or distribution list and that can also locate it in relation to the physical or organizational structure of MHS (or the network underlying it).

A.8 **attribute list**

F: liste d'attributes

S: lista de atributos

In the context of message handling, a data structure, an ordered set of attributes that constitutes an O/R address.

A.9 **attribute type**

F: type d'attribut

S: tipo de atributo

An identifier that denotes a class of information (e.g., personal names). It is a part of an attribute.

A.10 **attribute value**

F: valeur d'attribut

S: valor de atributo

An instance of the class of information an attribute type denotes (e.g., a particular personal name). It is a part of an attribute.

A.11 **basic service**

F: service de base

S: servicio básico

In the context of message handling, the sum of features inherent in a service.

A.12 **body**

F: corps

S: cuerpo

Component of a message. Other components are the heading and the envelope.

A.13 **body part**

F: partie du corps

S: parte del cuerpo

Component of the body of a message.

A.14 **common name**

F: nom courant

S: nombre común

In the context of message handling, a standard attribute of an O/R address form that identifies a user or distribution list relative to the entity denoted by another attribute (e.g., an organizational name).

A.15 **content**

F: contenu

S: contenido

In the context of message handling, an information object, part of a message, that the MTS neither examines nor modifies, except for conversion, during its conveyance of the message.

A.16 **content type**

F: type de contenu

S: tipo de contenido

In the context of message handling, an identifier, on a message envelope, that identifies the type (i.e. syntax and semantics) of the message content.

A.17 **conversion**

F: conversion

S: conversión

In the context of message handling, a transmittal event in which an MTA transforms parts of a message's content from one encoded information type to another, or alters a probe so it appears that the described messages were so modified.

A.18 **country name**

F: nom de pays

S: nombre de país

In the context of message handling, a standard attribute of a name form that identifies a country. A country name is a unique designation of a country for the purpose of sending and receiving messages.

Note – In the context of physical delivery additional rules apply (see also *physical delivery country name* and Recommendation F.415).

A.19 **delivery**

F: remise

S: entrega

In the context of message handling, a transmittal step in which an MTA conveys a message or report to the MS or UA of a potential recipient of the message or of the originator of the report's subject message or probe.

A.20 **delivery report**

F: rapport de remise

S: informe de entrega

In the context of message handling, a report that acknowledges delivery, non-delivery, export, or affirmation of the subject message or probe, or distribution list expansion.

A.21 **direct submission**

F: depôt direct

S: depósito directo

In the context of message handling, a transmittal step in which the originator's UA or MS conveys a message or probe to an MTA.

A.22 **direct user**

F: utilisateur direct

S: usuario directo

In the context of message handling, a user that engages in message handling by direct use of the MTS.

A.23 **directory**

F: annuaire

S: guía

A collection of open systems cooperating to provide directory services.

A.24 **directory name**

F: nom d'annuaire

S: nombre de guía

Name of an entry in a directory.

Note – In the context of message handling, the entry in the directory will enable the O/R address to be retrieved for submission of a message.

A.25 **directory system agent (DSA)**

F: agent de système d'annuaire (ASA)

S: agente de sistema de guía (ASG)

An OSI application process which is part of the directory, and whose role is to provide access to the directory information base to DUAs and/or other DSAs.

A.26 **directory user agent (DUA)**

F: agent d'usager d'annuaire (AUA)

S: agente de usuario de guía (AUG)

An OSI application process which represents a user in accessing the directoy. Each DUA serves a single user so that the directory can control access to directory information on the basis of the DUA names. DUAs can also provide a range of local facilities to assist users to compose requests (queries) and interpret the responses.

A.27 **distribution list (DL)**

F: liste de distribution (LD)

S: lista de distribución (LD)

In the context of message handling, the functional object, a component of the message handling environment, that represents a pre-specified group of users and other distribution lists and that is a potential destination for the information objects an MHS conveys.

Membership can contain O/R names identifying either users or other distribution lists.

A.28 **distribution list expansion**

F: allongement de liste de distribution

S: expansión de une lista de distribución

In the context of message handling, a transmittal event in which an MTA resolves a distribution list, among a message's immediate recipients, to its members.

A.29 **distibution list name**

F: nom de liste de distribution

S: nombre de lista de distribución

O/R name allocated to represent a collection of O/R addresses and directory names.

A.30 **domain**

F: domaine

S: dominio

See *management domain.*

A.31 **domain defined attributes**

F: attributs définis d'un domaine

S: atributos definidos por el dominio

Optional attributes of an O/R address allocated to names in the responsibility of a management domain.

A.32 **element of service**

F: élément de service

S: elemento de servicio

Functional unit for the purpose of segmenting and describing message handling features.

A.33 **encoded information type (EIT)**

F: type de codage (TC)

S: tipo de información codificada (TIC)

In the context of message handling, an identifier, on a message envelope, that identifies one type of encoded information represented in the message content. It identifies the medium and format (e.g., IA5 text, Group 3 facsimile) on an individual portion of the content.

A.34 **envelope**

F: enveloppe

S: sobre

In the context of message handling, an information object, part of a message, whose composition varies from one transmittal step to another and that variously identifies the message originator and potential recipients, documents its past and directs its subsequent conveyance by the MTS, and characterizes its content.

A.35 **explicit conversion**

F: conversion explicite

S: conversión explícita

In the context of message handling, a conversion in which the originator selects both the initial and final encoded information types.

A.36 **extension of physical delivery address components**

F: développement de composants d'adresse de remise physique

S: componentes de ampliación de dirección de entrega física

Standard attribute of a postal O/R address as a means to give further information about the point of physical delivery in a postal address, e.g., the name of a hamlet, or room and floor numbers in a large building.

A.37 **extension of postal O/R address components**

F: développement de composants d'adresse postale E/D

S: componentes de ampliación de dirección postal O/D

Standard attribute of a postal O/R address as a means to give further information to specify the addressee in a postal address, e.g. by organizational unit.

A.38 **formatted postal O/R address**

F: adresse postale E/D formatée

S: dirección postal O/D formatizada

O/R address based on a postal address with formatted attributes.

A.39 **heading**

F: en-tête

S: encabezamiento

Component of an IP-message. Other components are the envelope and the body.

A.40 **immediate recipient**

F: destinataire direct

S: destinatario inmediato

In the context of message handling, one of the potential recipients assigned to a particular instance of a message or probe (e.g., an instance created by splitting).

A.41 **implicit conversion**

F: conversion implicite

S: conversión implícita

In the context of message handling, a conversion in which the MTA selects both the initial and final encoded information types.

A.42 **indirect submission**

F: depôt indirect

S: depósito indirecto

In the context of message handling, a transmittal step in which an originator's UA conveys a message or probe to an MTA via an MS.

A.43 **indirect user**

F: utilisateur indirect

S: usuario indirecto

In the context of message handling, a user that engages in message handling by indirect use of MHS, i.e. through another communication system (e.g., a physical delivery system or the telex network) to which MHS is linked.

Note – Indirect users communicate via access units with direct users of MHS.

A.44 **intercommunication**

F: intercommunication

S: intercomunicación

In the context of message handling, a relationship between services where one of the services is a message handling service, enabling the user of the message handling service to communicate with users of other services.

Note – Examples are the intercommunication between the IPM service and the telex service, the intercommunication between message handling services and physical delivery services.

A.45 **interpersonal messaging service**

F: service de messagerie de personne à personne

S: servicio de mensajería interpersonal

Messaging service between users belonging to the same management domain or to different management domains by means of message handling, based on the message transfer service.

A.46 **IP-message**

F: message PP

S: mensaje IP

The content of a message in the IPM Service.

A.47 **local postal attributes**

F: attributs postaux locaux

S: atributos postales locales

Standard attributes of a post O/R address as a means to distinguish between places with the same name (e.g., by state name, county name, or geographical attribute) in a postal address.

A.48 **management domain (MD)**

F: domaine de gestion (DG)

S: dominio de gestión (DG)

In the context of message handling, a set of messaging systems – at least one of which contains, or realizes, an MTA – at that is managed by a single organization. It is a primary building block used in the organizational construction of MHS.

It refers to an organizational area for the provision of services.

Note – A management domain may or may not necessarily be identical with a geographical area.

A.49 **management domain name**

F: nom d'un domaine de gestion

S: nombre de dominio de gestión

Unique designation of a management domain for the purpose of sending and receiving messages.

A.50 **members**

F: membres

S: miembros

In the context of message handling, the set of users and distribution lists implied by a distribution list name.

A.51 **message**

F: message

S: mensaje

An instance of the primary class of information object conveyed by means of message transfer, and comprising an envelope and content.

A.52 **message handling (MH)**

F: messagerie (traitement des mesages) (M)

S: tratamiento de mensaje (TM)

A distributed information processing task that integrates the intrinsically related subtasks of message transfer and message storage.

A.53 **message handling environment**

F: environnement de traitement de messages

S: entorno de tratamiento de mensajes

The environment in which message handling takes place, comprising MHS, users, and distribution lists.

The sum of all components of message handling systems.

Note – Examples of components are:

- message transfer agents,
- user agents,
- message stores,
- access units,
- users.

A.54 **message handling service**

F: service de messagerie

S: servicio de tratamiento de mensajes

Service provided by the means of message handling systems.

Note 1 – Service may be provided through administration management domains or private management domains.

Note 2 – Examples of message handling services are:

- interpersonal messaging service (IPM service)
- message transfer service (MT service).

A.55 **message handling system (MHS)**

F: système de messagerie (STM)

S: sistema de tratamiento de mensajes (STM)

The functional object, a component of the message handling environment, that conveys information objects from one party to another.

A.56 **message storage**

F: mémorisation des messages

S: almacenamiento de mensajes

The automatic storage for later retrieval of information objects conveyed by means of message transfer. It is one aspect of message handling.

A.57 **message store (MS)**

F: mémoire des messages (MM)

S: memoria de mensajes (MM); alamacenador de mensajes (AM)

The functional object, a component of MHS, that provides a single direct user with capabilities for message storage.

A.58 **message transfer (MT)**

F: transfert de messages (TM)

S: transferencia de mensajes (TRM)

The non-real-time carriage of information objects between parties using computers as intermediaries. It is one aspect of message handling.

A.59 **message transfer agent (MTA)**

F: agent de transfert de messages (ATM)

S: agente de transferencia de mensajes (ATM)

A functional object, a component of the MTS, that actually conveys information objects to users and distribution lists.

A.60 **message transfer service**

F: service de transfert de messages

S: servicio de transferenicia de mensajes

Service that deals with the submission, transfer and delivery of messages for other messaging services.

A.61 **message transfer system (MTS)**

F: système de transfert de messages (système TM)

S: sistema de transferencia de mensajes (STRM)

The functional object consisting of one or more message transfer agents which provides store-and-forward message transfer between user agents, message stores and access units.

A.62 **messaging system**

F: système de messagerie

S: sistema de mensajeria

A computer system (possibly but not necessarily an open system) that contains, or realizes, one or more functional objects. It is a building block used in the physical construction of MHS.

A.63 **mnemonic O/R address**

F: adresse mnémonique E/D

S: dirección O/D nemotécnica

An O/R address tha mnemonicly identifies a user or distribution list relative to the ADMD through which the user is accessed or the distribution list is expanded. It identifies an ADMD, and a user or distribution list relative to that ADMD.

A.64 **naming authority**

F: autorité responsable de l'appellation

S: autoridad de denominación

An authority responsible for the allocation of names.

A.65 **network address**

F: adresse réseau

S: dirección de red

In the context of message handling, a standard attribute of an O/R address form that gives the network address of a terminal. It is comprising the numbering digits for network access points from an international numbering plan.

A.66 **non-delivery**

F: non-remise

S: no entrega

In the context of message handling, a transmittal event in which an MTA determines that the MTS cannot deliver a message to one or more of its immediate recipients, or cannot deliver a report to the originator of its subject message or probe.

A.67 **non-registered access**

F: accès non homologué

S: acceso no registrado

In the context of message handling services, access to the service through publicly available telecommunications means by users who have neither been explicitly registered by the service provider, nor been allocated an O/R address.

A.68 **numeric O/R address**

F: adresse numérique E/D

S: dirección O/D numérica

In the context of message handling, an O/R address that numerically identifies a user relative to the ADMD through which the user is accessed. It identifies an ADMD, and a user relative to that ADMD. It is identifying a user of message handling services by means of a numeric keypad.

A.69 **numeric user identifier**

F: identificateur numérique d'utilisateur

S: identificador de usuario numérico

Standard attribute of an O/R address as a unique sequence of numeric information for identifying a user.

A.70 **O/R address**

F: adresse E/D

S: dirección O/D

In the context of message handling, an attribute list that distinguishes one user or DL from another and identifies the user's point of access to MHS or the distribution list's expansion point.

A.71 **O/R name**

F: nom E/D

S: nombre O/D

In the context of message handling, an information object by means of which a user can be designated as the originator, or a user or distribution list designated as a potential recipient of a message or probe. An O/R name distinguishes one user or distribution list from another and can also identify its point of access to MHS.

A.72 **optional user facilities**

F: services complémentaires offerts en option á l'utilisateur

S: faciladad facultativa de usuario

In the context of message handling services the elements of service which are selectable by the user either on a contractual basis (agreed period of time) or on a per-message basis.

Note 1 – Optional user facilities are classified as either essential or additional.

Note 2 – Essential optional user facilities are to be made available to all message handling users.

Note 3 – Additional optional user facilities can be made available for national and international use on the basis of bilateral agreement between the service providers.

A.73 **organization name**

F: nom d'organisation

S: nombre de la organización

Standard attribute of an O/R address as a unique designation of an organization for the purpose of sending and receiving of messages.

A.74 **oganizational unit name**

F: nom d'une unité d'organisation

S: nombre de la unidad organizacional

Standard attribute of an O/R address as a unique designation of an organizational unit of an organization for the purpose of sending and receiving of messages.

A.75 **originator**

F: expediteur

S: originador

In the context of message handling, the user (but not distribution list) that is the ultimate source of a message or probe.

A.76 **personal name**

F: nom personnel

S: nombre personal

In the context of message handling, a standard attribute of an O/R address form that identifies a person relative to the entity denoted by another attribute (e.g., an organization name).

Note – Components are for example:

- surname,
- given name,
- initials,
- generation qualifier.

A.77 **physical delivery (PD)**

F: remise physique (RP)

S: entrega fisica (EF)

The delivery of a message in physical form, such as a letter, through a physical delivery system.

A.78 **physical delivery access unit (PDAU)**

F: unité d'accès de remise physique (UARP)

S: unidad de acceso de entrega física (UAEF)

An access unit that subjects messages (but neither probes nor reports) to physical rendition.

A.79 **physical delivery address components**

F: composants d'une adresse de remise physique

S: componentes de dirección de entrega física

In a postal address they contain the information necessary for the local physical delivery within the physical delivery area of the physical delivery office, i.e., a street address, a P.O. Box address, a poste restante address or a unique name alternatively.

Note – The information is generally restricted to one line with up to 30 printable graphic characters. Additional information may be supplied by using the attribute type "extension of physical delivery address components".

A.80 **physical delivery country name**

F: nom du pays de remise physique

S: nombre de país de entrega física

In the context of physical delivery, a unique description of the country of the final destination.

A.81 **physical delivery domain**

F: domaine de remise physique

S: dominio de entrega física

The domain of responsibility of an organization providing a physical delivery system and optionally an MTA/PDAU.

A.82 **physical delivery office address components**

F: composants d'une adresse de bureau de remise physique

S: componentes de dirección de oficina de entrega física

In a postal address they contain the information to specify the office which is responsible for the local physical delivery.

Note – The information is generally restricted to one line with up to 30 printable graphic characters. In some countries the postal code will follow the physical delivery office address components in a separate line (possibly together with the country name).

A.83 **physical delivery office name**

F: nom du bureau de remise physique

S: nombre de oficina de entrega física

Standard attribute of a postal O/R address, in the context of physical delivery, specifying the name of the city, village etc., where the physical delivery office is situated, or where the physical delivery is effected.

A.84 **physical delivery office number**

F: numéro du bureau de remise physique

S: número de oficina de entrega física

Standard attribute and in a postal O/R address a means to distinguish between more than one physical delivery office within a city etc.

A.85 **physical delivery organization name**

F: nom d'organisation de remise physique

S: nombre de la organización de entrega física

A free form name of the addressed entity in the postal address, taking into account the specified limitations in length.

A.86 **physical delivery personal name**

F: nom personnel de remise physique

S: nombre personal de entrega física

In a postal address a free form name of the addressed individual containing the family name and optionally the given name(s), the initial(s), title(s) and generation qualifier, taking into account the specified limitations in length.

A.87 **physical delivery service**

F: service de remise physique

S: servicio de entrega física

Service provided by a physical delivery system.

A.88 **physical delivery service name**

F: nom du service de remise physique

S: nombre del servicio de entrega física

Standard attribute of a postal O/R address in the form of the name of the service in the country electronically receiving the message on behalf of the physical delivery service.

A.89 **physical delivery system (PDS)**

F: système de remise physique (SRP)

S: sistema de entrega física (SEF)

A system that performs physical delivery. One important kind of physical delivery system is the postal system.

A.90 **physical message**

F: message physique

S: mensaje físico

A physical object comprising a relaying envelope and its content, e.g., a letter.

A.91 **physical rendition**

F: conversion physique

S: reproducción física

The transformation of an MHS message to a physical message, e.g., by printing the message on paper and enclosing it in a paper envelope.

A.92 **postal code**

F: code postal

S: código postal

Standard attribute of a postal O/R address to specify the geographical area, and in the context of MHS, used for routing of messages.

A.93 **postal O/R address**

F: adresse postale E/D

S: dirección postal O/D

In the context of message handling, an O/R address that identifies a user by means of its postal address. It identifies the physical delivery system through which the user is to access and gives the user's postal address.

A.94 **postal O/R address components**

F: composants d'une adresse postale E/D

S: componentes de dirección postal O/D

They contain in a postal address information to decribe the sender or addressee by means of his name (physical delivery personal name, physical delivery organization name).

Note – In a postal address the information is generally restricted to one line of 30 printable characters. Additional information may be supplied by using the attribute type "extension of postal O/R address components".

A.95 **post office box address (P.O. box address)**

F: adresse de case postale

S: dirección-apartado de correos

An access unit that subjects messages (but neither probes nor reports) to physical rendition.

A.96 **post restante address**

F: adresse poste restante

S: dirección lista de correos

A standard attribute in a postal address indicating that physical delivery at the counter is requested. It may also carry a code.

A.97 **potential recipient**

F: destinataire potential

S: destinatario potencial

In the context of message handling, any user or distribution list to which a message or probe is conveyed during the course of tansmittal. Equivalently, a preferred member, alternate member, or substitute recipient.

A.98 **preferred recipient**

F: destinataire préféré

S: receptor preferido

In the context of message handling, one of the users and distribution lists that the originator selects as a message's or probe's preferred destination.

A.99 **private domain name**

F: nom d'un domain privé

S: nombre de dominio privado

In the context of message handling, a standard attribute of an O/R address form that identifies a PRMD relative to the ADMD denoted by an administration domain name.

Note – They are administered by the ADMD the PRMD is associated with.

A.100 **private management domain (PRMD)**

F: domaine de gestion privé (DGPR)

S: dominio de gestión privado (DGPR)

In the context of message handling, a management domain that comprises messaging system(s) managed by an organization other than an Administration.

A.101 **probe**

F: essai

S: sonda

In the context of message handling, an instance of a secondary class of information objects conveyed by means of message transfer that describes a class of message and that is used to determine the deliverability of such messages.

A.102 **public message handling service**

F: service public de messagerie

S: servicio público de tratamiento de mensajes

Message handling service offered by an Administration.

A.103 **public services**

F: services publics

S: servicios públicos

In the context of telecommunication, the services offered by Administrations.

A.104 **receipt**

F: réception

S: recepción

In the context of message handling, a transmittal step in which either a UA conveys a message or report to its direct user, or the communication system that serves an indirect user conveys such an information object to that user.

A.105 **recipient**

F: destinataire

S: destinatario

See *actual recipient.*

A.106 **recursion**

F: récursivité

S: repetición

In the context of message handling, the situation that a message gets back to the same distribution list of origin and potentially circulates infinitely.

A.107 **redirection**

F: réacheminement

S: redireccionamiento

In the context of message handling, a transmittal event in which an MTA replaces a user among a message's immediate recipients with a user preselected for that message.

A.108 **registered access**

F: accès homologué

S: acceso registrado

In the context of message handling services, access to the service performed by subscribers who have been registered by the service provider to use the service, and been allocated an O/R address.

A.109 **report**

F: rapport

S: informe

In the context of message handling, an instance of a secondary class of information object conveyed by means of message transfer. It is generated by the MTS, it reports the outcome or progress of a message's or probe's transmittal to one or more potential recipients.

A.110 **retrieval**

F: extraction

S: recuperación

In the context of message handling, a transmittal step in which a user's message store conveys a message or report to the user's UA. The user is an actual recipient of the message or the originator of the subject message or probe.

A.111 **security capabilities**

F: capacité de sécurité

S: capacidades de seguridad

In the context of message handling, the mechanisms that protect against various security threats.

A.112 **specialized access**

F: accès spécialisé

S: acceso especializado

In the context of message handling, the involvement of specialized access units providing intercommunication between message handling services and other telecommunication services.

A.113 **standard attribute**

F: attribut normalisé

S: atributo normalizado

An attribute whose type is bound to a certain class of information.

A.114 **street address**

F: adresse de rue

S: dirección-calle

A standard attribute in a postal address giving information for the local distribution and physical delivery, i.e. the street name, the street identifier (like street, place, avenue) and the house number.

A.115 **subject**

F: objet

S: asunto

In the context of message handling, the information, part of the header, that summarizes the content of the message as the originator has specified it.

A.116 **subject message**

F: message objet

S: mensaje de asunto

The message that is the subject of a report.

A.117 **subject probe**

F: essai objet

S: sonda de asunto

The probe that is the subject of a report.

A.118 **submission**

F: dépôt

S: depósito

Direct submission or indirect submission.

A.119 **substitute recipient**

F: destinataire substitut

S: destinatario sustituto

In the context of message handling, the user or distribution list to which a preferred, alternate, or member (but not another substitute) recipient can have elected to redirect messages (but not probes).

A.120 **terminal identifier**

F: identificateur de terminal

S: identificador de terminal

Standard attribute in an O/R address providing information for identifying a terminal amongst others.

Note – Examples are telex answerback and teletex terminal identifier.

A.121 **terminal O/R address**

F: adresse terminale E/D

S: dirección O/D de terminal

In the context of message handling, an O/R address that identifies a user by means of the network address of his terminal and that can identify the ADMD through which that terminal is accessed. The terminals identified can belong to different networks.

A.122 **terminal type**

F: type de terminal

S: tipo de terminal

Standard attribute of an O/R address that indicates the type of a terminal.

Note – Examples: telex, teletex, G3 facsimile, G4 facsimile, IA5, videotex terminal.

A.123 **transfer**

F: transfert

S: transferencia

In the context of message handling, a transmittal step in which one MTA conveys a message, probe, or report to another.

A.124 **transfer system**

F: système de transfert

S: sistema de transferencia

A messaging system that contains one MTA; optionally one or more access units, and neither a UA nor a message store.

A.125 **transmittal**

F: transmission

S: transmisión

The conveyance or attempted conveyance of a message from its originator to its potential recipients, or of a probe from its originator to MTAs able to affirm any described message's deliverability to its potential recipients. It also encompasses the conveyance or attempted conveyance, to the originator of the message or probe, or any report it provokes. It is a sequence of transmittal steps and events.

A.126 **unformatted postal O/R address**

F: adresse postale E/D non formatée

S: dirección postal O/D no formatizada

O/R address based on an unformatted postal address.

A.127 **unique postal name**

F: nom postal unique

S: nombre postal exclusivo

In a postal address a standard attribute describing the point of physical delivery by means of a unique name, e.g. that of a building.

A.128 **user**

F: usager/utilisateur

S: usuario

In the context of message handling, a functional object (e.g., a person), a component of the message handling environment, that engages in (rather than provides) message handling and that is a potential source or destination for the information objects an MHS conveys.

A.129 **user agent (UA)**

F: agent d'usager (AU)

S: agente de usuario (AU)

In the context of message handling, the functional object, a component of MHS, by means of which a single direct user engages in message handling.

Component of MHS the user interacts with.

ANNEX B

(to Recommendation X.400)

Definitions of elements of service

Note – The abbreviations used in the title lines have the following meanings:

TM Message transfer

IPM Interpersonal messaging

PD Physical delivery

MS Message store

PR Per recipient (available on a per-recipient basis)

B.1 *Access management* TM

This element of service enables a UA and MTA to establish access to one another and to manage information associated with access establishment.

The element of service permits the UA and MTA to identify and validate the identity of the other. It provides a capability for the UA to specify its O/R address and to maintain access security. When access security is achieved through passwords, these passwords can be periodically updated.

Note – A more secure form of access management is provided by the element of service secure access management.

B.2 *Additional physical rendition* PD PR

This element of service allows an originating user to request the PDAU to provide the additional renditon facilities (e.g., kind of paper, colour printing, etc.). Bilateral agreement is required to use this element of service.

B.3 *Alternate recipient allowed* MT

This element of service enables an originating UA to specify that the message being submmitted can be delivered to an alternate recipient as described below.

A destination MD will interpret all of the user attributes in order to select a recipient UA. Three cases can be distinguished:

1) all the attributes match precisely those of a subscriber UA. Delivery is attempted to that UA;
2) either insufficient attributes are supplied or those supplied match those of more than one subscriber UA. The message cannot be delivered;
3) at least the minimum set of attributes required by the destination MD is supplied. Nevertheless, taking all of the other attributes into account, the attributes match those of no UA.

In case 3, an MD that supports the alternate recipient assignment element of service can deliver the message to a UA that has been assinged to receive such messages. This UA will be notified of the O/R address of the intended recipient as specified by the originator. Delivery to this UA will be reported in a delivery notification if requested by the originator.

B.4 *Alternate recipient assignment* MT

This element of service enables a UA to be given the capability to have certain messages delivered to it for which there is not an exact match between the recipient attributes specified and the name of the user. Such a UA is specified in terms of one or more attributes for which an exact match is required, and one or more attributes for which any value is acceptable. For example, an orginization can establish a UA to receive all messages for which country name, administration management domain name and organization name (for example, company name) are an exact match but the personal name of the recipient does not correspond to an individual known by an MHS in that organization. This permits the organization to manually handle the messages to these individuals.

In order for a message to be reassigned to an alternate recipient, the originator must have requested the alternate recipient allowed element of service.

B.5 *Authorizing users indication* IPM

This element of service allows the originator to indicate to the recipient the names of the one or more persons who authorized the sending of the message. For example, an individual can authorize a particular action which is subsequently communicated to those concerned by another person such as a secretary. The former person is said to authorize its sending while the latter person is the one who sent the message (originator). This does not imply signature-level authorization.

B.6 *Auto-forwarded indication* IPM

This element of service allows a recipient to determine that a body of an incoming IP-message contains an IP-message that has been auto-forwarded. Thus the recipient can distinguish from that where an incoming IP-message contains a forwarded message (as described in § B-31) in the body. As with a forwarded IP-message, an auto-forwarded IP-message can be accompanied by information (for example, time stamps, indication of conversion) associated with its original delivery.

Note – The indication that auto-forwarding of an IP-message has occurred enables a recipient IPM UA, should it so choose, to prevent further auto-forwarding and thus the possibility of loops. In addition, a recipient IPM UA can choose whether or not to auto-forward based on other criteria (for example, sensitivity classification).

When an IPM UA auto-forwards an IP-message, it designates it as auto-forwarded. If receipt/non-receipt notification has been requested for the IP-message being auto-forwarded, the IPM UA generates a non-receipt notification informing the originator of the auto-forwarding of the IP-message. The notification optionally includes a comment supplied by the originally intended recipient. No further notification applying to the auto-forwarded IP-message is generated by any IPM UA.

B.7 *Basic physical rendition* PD PR

This element of service enables the PDAU to provide the basic renditon facilities for converting the MHS message into a physical message. This is the default action to be taken by the PDAU.

B.8 *Blind copy recipient indication* IPM PR

This element of service allows the originator to provide the O/R name of one or more additional users, or DLs, who are intended recipients of the IP-message being sent. These names are not disclosed to either the primary or copy recipients. Whether or not these additional recipients are disclosed to one another is a local matter.

B.9 *Body part encryption indication* IPM

This element of service allows the originator to indicate to the recipient that a particular body of the IP-message being sent has been encrypted. Encryption can be used to prevent unauthorized inspection or modification of the body part. This element of service can be used by the recipient to determine that some body part(s) of the IP-message must be decrypted. This element of service, however, does not itself encrypt or decrypt any body part.

B.10 *Content confidentiality* MT

This element of service allows the originator of a message to protect the content of the message from disclosure to recipients other than the intended recipient(s). Content confidentiality is on a per-message basis, and can use either an asymmetric or a symmetric encryption technique.

B.11 *Content integrity* MT PR

This element of service allows the originator of the message to provide to the recipient of the message a means by which the recipient can verify that the content of the message has not been modified. Content integrity is on a per-recipient basis, and can use either an asymmetric or a symmetric encryption technique.

B.12 *Content type indication* MT

This element of service enables an originating UA to indicate the content type for each submitted message. A recipient UA can have one or more content types delivered to it. An example of a content type is the contents generated by the IPM class of cooperating UAs.

B.13 *Conversion prohibition* MT

This element of service enables an originating UA to instruct the MTS that implicit encoded information type conversion(s) should not be performed for a particular submitted message.

B.14 *Conversion prohibition in case of loss of information* MT

This element of service enables an originating UA to instruct the MTS that encoded information type conversion(s) should not be performed for a particular submitted message if such conversion(s) would result in loss of information. Loss of information is discussed in detail in X.408.

Should this and the conversion prohibition element of service both be selected, the latter shall take precedence.

Note – This element of service will not protect against possible loss of information in certain cases where the recipient is using an I/O device whose capabilities are unknown to the MTA.

B.15 *Converted indication* MT PR

This element of service enables the MTS to indicate to a recipient UA that the MTS performed encoded information type conversion on a delivered message. The recipient UA is informed of the resulting types.

B.16 *Counter collection* PD PR

This element of service allows an originating user to instruct the PDS to keep the physical message ready for counter collection at the post office specified by the originator, or at the post office which offers counter collection service closest to the given recipient's address.

B.17 *Counter collection with advice* PD PR

This element of service allows an originating user to instruct the PDS to keep the physical message ready for counter collection at the post office specified by the originator, or at the post office which offers counter collection service closest to the given recipient's address, and to inform the recipient via telephone, or telex, or teletex, using the number provided by the originator.

B.18 *Cross-referencing indication* IPM

This element of service allows the originator to associate with the IP-message being sent, the globally unique identifiers of one or more other IP-messages. This enables the recipient's IPM UA, for example, to retrieve from storage a copy of the referenced IP-messages.

B.19 *Deferred delivery* MT

This element of service enables an originating UA to instruct the MTS that a message being submitted shall be delivered no sooner than a specified data and time. Delivery will take place as close to the date and time specified as possible, but not before. The date and time specified for deferred delivery is subject to a limit which is defined by the originator's management domain.

Note – Storage of the message shall be handled in the originating country.

B.20 *Deferred delivery cancellation* MT

This element of service enables an originating UA to instruct the MTS to cancel a previously submitted deferred delivery message. The cancellation attempt may or may not always succeed. Possible reasons for failure are: deferred delivery time has passed, or the message has already been forwarded within the MTS.

B.21 *Delivery notification* MT PR

This element of service enables an originating UA to request that the originating UA be explicitly notified when a submitted message has been successfully delivered to a recipient UA or access unit. The notification is related to the submitted message by means of the message indentifier and includes the date and time of delivery. In the case of a multi-destination message, the originating UA can request this element of service or a per-recipient basis.

When a message is delivered after distribution list expansion, then, depending on the policy of the distribution list, the notification can be sent to either the list owner, the message originator, or both.

Delivery notification carries no implication that any UA or user action, such as examination of the message content, has taken place.

B.22 *Delivery time stamp indication* MT PR

This element of service enables the MTS to indicate to a recipient UA the date and time at which the MTS delivered a message. In the case of physical delivery, this element of service indicates the date and time at which the PDAU has taken responsibility for printing and further delivery of the physical message.

B.23 *Delivery via bureaufax service* PD PR

This element of service allows an originating user to instruct the PDAU and associated PDS to use the bureaufax service for transport and delivery.

B.24 *Designation of recipient by directory name* MT PR

This element of service enables an originating UA to use a directory name in place of an individual recipient's O/R address.

B.25 *Disclosure of other recipients* MT

This element of service enables the originating UA to instruct the MTS then submitting a multi-recipient message, to disclose the O/R names of all other recipients to each recipient UA, upon delivery of the message. The O/R names disclosed are as supplied by the originating UA. If distribution list expansion has been performed, then only the originator specified DL name will be disclosed, and not the names of its members.

B.26 *DL expansion history indication* MT

This element of service provides to a recipient, at delivery, information about the distribution list(s) through which the message has arrived. It is a local matter as to how much of this information is presented to the recipient.

B.27 *DL expansion prohibited* MT

This element of service allows an originating user to specify that if any of the recipients can directly or via reassignment refer to a distribution list, then no expansion shall occur. Instead, a non-delivery notification will be returned to the originating UA, unless prevention of non-delivery notification has been requested.

B.28 *EMS (express mail service)* PD PR

This element of service allows an originating user to instruct the PDS to transport and deliver the physical message produced from the MHS message through accelerated letter circulation and delivery service (such as EMS or the equivalent domestic service) in the destination country.

B.29 *Expiry date indication* IPM

This element of service allows the originator to indicate to the recipient the date and time after which he considers the IP-message to be invalid. The intent of this element of service is to state the originator's assessment of the current applicability of an IP-message. The particular action on behalf of a recipient by his IPM UA, or by the recipient himself, is unspecified. Possible actions might be to file or delete the IP-message after the expiry date has passed.

B.30 *Explicit conversion* MT PR

This element of service enables an originating UA to request the MTS to perform a specified conversion, such as required when interworking between different telematic services. When a message is delivered after conversion has been performed, the recipient UA is informed of the original encoded information types as well as the current encoded information types in the message.

Note 1 – This element of service is intended to support interworking with telematic terminals/services.

Note 2 – When DL names are used in conjunction with this element of service, conversion will apply to all members of the DL.

B.31 *Forwarded IP-message indication* IPM

This element of service allows a forwarded IP-message, or a forwarded IP-message plus its "delivery information" to be sent as the body (or as one of the body parts) of an IP-message. An indication that the body part is forwarded is conveyed along with the body part. In a multi-part body, forwarded body parts can be included along with body parts of other types. "Delivery information" is information which is conveyed from the MTS when an IP-message is delivered (for example, time stamps and indication of conversion). However, inclusion of this delivery information along with a forwarded IP-message in no way guarantees that this delivery information is validated by the MTS.

The receipt notification request indication and the non-receipt notification request elements of service are not affected by the forwarding of a IP-message.

B.32 *Grade of delivery selection* MT

This element of service enables an originating UA to request that transfer through the MTS be *urgent* or *non-urgent*, rather than *normal*. The time periods defined for non-urgent and urgent transfer are longer and shorter, respectively, than that defined for normal transfer. This indication is also sent to the recipient with the message.

B.33 *Hold for delivery* MT

This element of service enables a recipient UA to request that the MTS hold its messages and returning notifications for delivery until a later time. The UA can indicate to the MTS when it is unavailable to take delivery of messages and notifications, and also, when it is again ready to accept delivery of messages and notifications from the MTS. The MTS can indicate to the UA that messages are waiting due to the criteria the UA established for holding messages. Responsibility for the management of this element of service lies with the recipient MTA.

Criteria for requesting a message to be held for delivery are: encoded information type, content type, maximum content length, and priority. The message will be held until the maximum delivery time for that message expires, unless the recipient releases the hold prior to its expiry.

Note – The hold for delivery element of service is distinct from the message store facility. The hold for delivery element of service provides temporary storage to facilitate delivery and only after a message has been transferred to the recipient's UA, is delivery notification returned. The message store facility augments the storage of a UA and can be used to store messages for an extended period of time. Unlike the hold for delivery element of service, delivery notifications are returned as soon as the message is placed in (that is, delivered to) the message store.

B.34 *Implicit conversion* MT

This element of service enables a recipient UA to have the MTS perform for a period of time any necessary conversion on messages prior to delivery. Neither the originating nor recipient UA explicitly requests this element of service on a per-message basis. If the encoded information type capabilities of the recipient UA are such that more than one type of conversion can be performed, the most appropriate conversion is performed. When a message is delivered after conversion has been performed, the recipient UA is informed of the original encoded information types as well as the current encoded information types in the message.

B.35 *Importance indication* IPM

This element of service allows the originator to indicate to the recipients his assessment of the importance of the IP-message being sent. Three levels of importance are defined: *low*, *normal*, and *high*.

This element of service is not related to the grade of delivery selection element of service provided by the MTS. The particular action taken by the recipient or his IPM UA based on the importance categorization is unspecified. It is the intent to allow the recipient IPM UA, for example, to present IP-messages in order of their importance or to alert the recipient of the arrival of IP-messages of high importance.

B.36 *Incomplete copy indication* IPM

This element of service allows an originator to indicate that this IP-message is an incomplete copy of an IP-message with the same IP-message identification in that one or more body parts, and/or heading fields of the original IP-message are absent.

B.37 *IP-message identification* IPM

This element of service enables cooperating IMP UAs to convey a globally unique identifier for each IP-message sent or received. The IP-message identifier is composed of an O/R name of the originator and an identifier that is unique with respect to that name. IPM UAs and users use this identifier to refer to a previously sent or received IP-message (for example, in receipt notifications).

B.38 *Language indication* IPM

This element of service enables an originating UA to indicate the language type(s) of a submitted IP-message.

B.39 *Latest delivery designation* MT

This element of service enables an originating UA to specify the latest time by which the message is to be delivered. If the MTS cannot deliver by the time specified, the message is not delivered and is cancelled. On multi-recipient messages, the latest delivery time can expire prior to delivery to all recipients, but this will not negate any deliveries which have already occurred.

B.40 *Message flow confidentiality* MT

This element of service allows the originator of the message to protect information which might be derived from observation of the message flow.

Note – Only a limited form of this is supported.

B.41 *Message identification* MT

This element of service enables the MTS to provide a UA with a unique identifier for each message or probe submitted or delivered by the MTS. UAs and the MTS use this identifier to refer to a previously submitted message in connection with elements of service such as delivery and non-delivery notification.

B.42 *Message origin authentication* MT PR

This element of service allows the originator of a message to provide to the recipient(s) of the message, and any MTA through which the message is transferred, a means by which the origin of the message can be authenticated (i.e. a signature). Message origin authentication can be provided to the recipient(s) of the message, and any MTA through which the message is transferred, on a per-message basis using an asymmetric encryption technique, or can be provided only to the recipient(s) of the message, on a per-recipient basis using either an asymmetric or a symmetric encryption technique.

B.43 *Message security labelling* MT

This element of service allows the originator of a message (or probe) to associate with the message (and any reports on the message or probe) an indication of the sensitivity of the message (a security label). The message security label may be used by the MTS and the recipient(s) of the message to determine the handling of the message in line with the security policy in force.

B.44 *Message sequence integrity* MT PR

This element of service allows the originator of the message to provide to a recipient of the message a means by which the recipient can verify that the sequence of messages from the originator to the recipient has been preserved (without message loss, re-ordering, or replay). Message sequence integrity is on a per-recipient basis, and can use either an asymmetric or a symmetric encryption technique.

B.45 *Multi-destination delivery* MT PR

This element of service enables an originating UA to specify that a message being submitted is to be delivered to more than one recipient UA. Simultaneous delivery to all specified UAs is not implied by this element of service.

B.46 *Multi-part body* IPM

This element of service allows an originator to send to a recipient or recipients an IP-message with a body that is partitioned into several parts. The nature and attributes, or type, of each body part are conveyed along with the body part.

B.47 *Non-delivery notification* MT PR

This element of service enables the MTS to notify an originating UA if a submitted message was not delivered to the specified recipient UA(s). The reason the message was not deliverd is included as part of the notification. For example, the recipient UA can be unknown to the MTS.

In the case of a multi-destination message, a non-delivery notification can refer to any or all of the recipient UAs to which the message could not be delivered.

When a message is not delivered after distribution list expansion, then, depending on the policy of the distribution list, the notification can be sent to either the list owner, the message originator, or both.

B.48 *Non-receipt notification request indication* IPM PR

This element of service allows the originator to ask that he be notified should the IP-message be deemed unreceivable. In the case of a multi-recipient IP-message, the originator can request this element of service on a per-recipient basis.

The originator's UA conveys his request to the recipient's UA. The recipient's UA automatically issues a non-recipient notification when any of the following events occur:

1) the recipient's UA auto-forwards the IP-message to another user;
2) the recipient's UA discards the IP-message prior to receipt;
3) the recipient's subscription is terminated before he receives the IP-message.

Since receipt can occur arbitrarily long after delivery, the recipient's failure to access the IP-message, even for a long period of time (for example, while on an extended business trip), does not constitute non-receipt and thus no notification is issued.

Note – No legal significance can be adduced from this element of service.

B.49 *Non-repudiation of delivery* MT PR

This element of service allows the originator of a message to obtain from the recipient(s) of the message irrevocable proof that the message was delivered to the recipient(s). This will protect against any attempt by the recipient(s) to subsequently deny receiving the message or its content. Non-repudiation of delivery is provided to the originator of a message on a per-recipient basis using asymmetric encryption techniques.

B.50 *Non-repudiation of origin* MT PR

This element of service allows the originator of a message to provide the recipient(s) of the message irrevocable proof of the origin of the message. This will protect against any attempt by the originator to subsequently revoke the message or its content. Non-repudiation of origin is provided to the recipient(s) of a message on a per-message basis using asymmetric encryption techniques.

B.51 *Non-repudiation of submission* MT

This element of service allows the originator of a message to obtain irrevocable proof that a message was submitted to the MTS for delivery to the originally specified recipient(s). This will protect against any attempt by the MTS to subsequently deny that the message was submitted for delivery to the originally specified recipient(s). Non-repudiation of submission is provided to the originator of a message on a per-message basis, and uses an asymmetric encryption technique.

B.52 *Obsoleting indication* IPM

This element of service allows the originator to indicate to the recipient that one or more IP-messages he sent previously are obsolete. The IP-message that carries this indication supersedes the obsolete IP-message.

The action to be taken by the recipient or his IPM UA is a local matter. The intent, however, is to allow the IPM UA or the recipient to, for example, remove or file obsolete messages.

B.53 *Ordinary mail* PD PR

This element of service enables the PDS to transport and deliver the letter produced from the MHS message in the mode available through the ordinary letter mail service in the country of destination. This is the default action for the transport and delivery of a physical message.

B.54 *Original encoded information types indication* MT

This element of service enables an originating UA to specify to the MTS the encoded information types of a message being submitted. When the message is delivered, it also indicates to the recipient UA the encoded information types of the message specified by the originating UA.

B.55 *Orinigator indication* IPM

This element of service allows the identity of the originator to be conveyed to the recipient. The intent of this IPM element of service is to identify the originator in a user-friendly way. In contrast, the MTS provides to the recipient the actual O/R address and directory name, if present, of the originator. DL names should not be used in originator indication.

B.56 *Originator requested alternate recipient* MT PR

This element of service enables an originating UA to specify, for each intended recipient, one alternate recipient to which the MTS can deliver the message, if delivery to the intended recipient is not possible. The alternate recipient can be a distribution list. For the purposes of determining success or failure (and hence delivery and non-delivery notifications), delivery to the originator requested alternate recipient is equivalent to delivery to the intended recipient. If the intended recipient has requested redirection of incoming messages, and if the originating UA has requested redirection allowed by the originator, the system first tries to redirect the message. If this fails, the system then attempts to deliver the message to the designated alternate recipient.

B.57 *Physical delivery notification by MHS* PD PR

This element of service allows an originating user to request that an explicit notification, informing the originator of either successful or unsuccessful delivery of the physical message, be generated and returned by MHS. The notification provides information on delivery but no physical record is provided by the PDS.

Note 1 – The notification includes the date and time of delivery based on the delivery confirmation given by the delivery person, the addressee or another authorized person. This is subject to national regulations in the destination contry and is also dependent on the type of delivery requested (e.g., in the case of registered mail to addressee in person, the addressee would be the confirming person).

Note 2 – This notification carries no implication that any action on the part of the recipient (such as examination of the message content) has taken place.

Note 3 – When this element of service is requested, and the physical message is undeliverable, it is either returned or destroyed depending on national regulations in the destination country, which means that the default action of the element of service B.91 is overridden.

B.58 *Physical delivery notification by PDS* PD PR

This element of service allows an originating user to request that an explicit notification, informing the originator of either successful or unsuccessful delivery of the physical message, be generated and returned by the PDS. The notification serves as a record of delivery for the originating user to retain for reference.

Note 1 – The notification includes the date and time, and, in the case of successful delivery, the signature of the person confirming the delivery. The confirming person can be the delivery person, the addressee or another authorized person. This is subject to national regulations in the destination country and is also dependent on the type of delivery requested (e.g., in the case of registered mail to addressee in person, the addressee would be the confirming person).

Note 2 – This notification carries no implication that any action on the part of the recipient (such as examination of the message content) has taken place.

Note 3 – When this element of service is requested, and the physical message is undeliverable, is either returned or destroyed depending on national regulations in the destination country, which means that the default action of the element of service B.91 is overridden.

B.59 *Physical forwarding allowed* PD PR

This element of service enables the PDS to forward the physical message to a forwarding address if the recipient has changed his address and indicated this to the PDS. This is the default action taken by the PDS.

B.60 *Physical forwarding prohibited* PD PR

This element of service allows an originating user to instruct the PDS not to forward the physical message to a forwarding address.

B.61 *Prevention of non-delivery notification* MT PR

This element of service enables an originating UA to instruct the MTS not to return a non-delivery notification to the originating UA in the event that the message being submitted is judged undeliverable. In the case of a multi-destination message, the originating UA can request this element of service on a per-recipient basis.

B.62 *Primary and copy recipients indication* IPM

This element of service allows the originator to provide the names of zero or more users, or DLs, who are the intended primary recipients of the IP-message, and the names of zero or more users, or DLs, who are the intended copy recipients of the IP-message. It is intended to enable a recipient to determine the category in which each of the specified recipients (including the recipient himself) was placed. The exact distinction between these two categories of recipients is unspecified. However, the primary recipients, for example, might be expected to act upon the IP-message, while the copy recipients might be sent the IP-message for information only.

Note – As an example of this element of service in a typical memorandum, the primary recipients are normally designated by the directive "to:" while "cc:" identifies the copy recipients.

B.63 *Probe* MT

This element of service enables a UA to establish before submission whether a particular message could be delivered. The MTS provides the submission information and generates delivery and/or non-delivery notifications indicating whether a message with the same submission information could be delivered to the specified recipient UAs.

The probe element of service includes the capability of checking whether the content size, content type, and/or encoded information types would render it undeliverable. The significance of the result of a probe depends upon the recipient UA(s) having registered with the MTS the encoded information types, content type and maximum message size that it can accept. This element of service is subject to the same delivery time targets as for the urgent class. In the case of DLs, a probe indicates nothing about the likelihood of successful delivery to the DL members, but only whether the originator has the right to submit to the DL.

B.64 *Probe origin authentication* MT

This element of service allows the originator of a probe to provide to any MTA through which the probe is transferred a means to authenticate the origin of the probe (i.e. a signature). Probe origin authentication is on a per-probe basis, and uses an asymmetric encryption technique.

B.65 *Proof of delivery* MT PR

This element of service allows the originator of a message to obtain from the recipient(s) of the message the means to authenticate the identity of the recipient(s) and the delivered message and content. Message recipient authentication is provided to the originator of a message on a per-recipient basis using either symmetric or asymmetric encryption techniques.

B.66 *Proof of submission* MT

This element of service allows the originator of a message to obtain from the MTS the means to authenticate that the message was submitted for delivery to the originally intended recipient. Message submission authentication is provided on a per-message basis, and can use symmetric or asymmetric encryption techniques.

B.67 *Receipt notification request indication* IPM PR

This element of service allows the originator to ask that he be notified when the IP-message being sent is received. In the case of a multi-recipient message, the originator can request this element of service on a per-recipient basis. This element of service also implicitly requests non-receipt notification request indication.

The originator's UA conveys his request to the recipient's UA. The recipient can instruct his UA to honour such requests, either automatically (for example, when it first renders the IP-message on the recipient's terminal) or upon his explicit command. The recipient can also instruct his UA, either in blanket fashion or case by case, to ignore such requests.

B.68 *Redirection disallowed by originator* MT

This element of service enables an originating UA to instruct the MTS, if the recipient has requested the redirection of incoming messages element of service, that redirection should not be applied to a particular submitted message.

B.69 *Redirection of incoming messages* MT

This element of service enables a UA to instruct the MTS to redirect incoming messages addressed to it, to another UA or to a DL, for a specified period of time, or until revoked.

Note 1 – This is an MT element of service that does not necessitate delivery to the intended recipient before redirection can take place. It is therefore distinct from the IPM Auto-Forwarded Indication Element of Service.

Note 2 – When security provisions are in force, different incoming messages, on the basis of their security labels, may be redirected to separate alternate recipients or not re-directed at all.

B.70 *Registered mail* PD PR

This element of service allows an originating user to instruct the PDS to handle the physical message as registered mail.

B.71 *Registered mail to addressee in person* PD PR

This element of service allows an originating user to instruct the PDS to handle the physical message as registered mail and to deliver it to the addressee only.

B.72 *Reply request indication* IPM PR

This element of service allows the originator to request that a recipient send an IP-message in reply to the IP-message that carries the request. The originator can also specify the date by which any reply should be sent, and the one or more users and DLs to whom the originator requests (but does not demand) be among the preferred recipients of any reply. The recipient is informed of the date and names but it is up to the recipient to decide whether or not, and if so, to whom to reply.

Note – A blind copy recipient should consider carefully to whom he sends a reply, in order that the meaning of the blind copy recipient indication element of service is preserved.

B.73 *Replying IP-message indication* IPM

This element of service allows the originator of an IP-message to indicate to the recipient(s) that this IP-message is being sent in reply to another IP-message. A reply can, depending on the wishes of the originator of the replied-to message, and the final decision of the originator of the reply, be sent to:

1) the recipients specified in the reply request indication of the replied-to message;
2) the originator of the replied-to message;
3) the originator and other recipients;
4) a distribution list, in which the originator of the replied-to message can be a receiving member;
5) other recipients as chosen by the originator of the reply.

The recipients of the reply receive it as a regular IP-message, together with an indication of which IP-message it is a reply to.

B.74 *Report origin authentication* MT

This element of service allows the originator of a message (or probe) to authenticate the origin of a report on the delivery or non-delivery of the subject message (or probe), (a signature). Report origin authentication is on a per-report basis, and uses an asymmetric encryption technique.

B.75 *Request for forwarding address* PD PR

This element of service allows an originating user to instruct the PDS to provide the forwarding address if the recipient has changed his address and indicated this to the PDS.

This element of service can be used with either physical forwarding allowed or prohibited. The provision of the forwarding address by the PDS to an originating user is subject to national regulations in the destination country. The default action is no provision of the forwarding address.

B.76 *Requested delivery method* MT PR

This element of service allows a user to request, on a per-recipient basis, the preference of method or methods of message delivery (such as through an access unit). Non-delivery results if preference(s) cannot be satisfied.

B.77 *Restricted delivery* MT

This element of service enables a recipient UA to indicate to the MTS that it is not prepared to accept delivery of messages from certain originating UAs or DLs.

Note 1 – This element of service can be requested in either of two ways:

a) specification by the recipient UA of unauthorized originators, all other originators are considered as authorized;
b) specification by the recipient UA of authorized originators, all other originators are considered to be unauthorized.

Note 2 – The MTS abstract service specified in Rec. X.411 does not provide a technical realization of this element of service. Its provision may be the subject of further standardization.

B.78 *Return of content* MT

This element of service enables an originating UA to request that the content of a submitted message be returned with any non-delivery notification. This will not be done, however, if any encoded information type conversion has been performed on the message's content.

B.79 *Secure access management* MT

This element of service enables an MTS user to establish an association with the MTS, or the MTS to establish an association with an MTS user, or an MTA to establish an association with another MTA. It also establishes the strong credentials of the objects to interact, and the context and security-context of the association. Secure access management can use either an asymmetric or a symmetric encryption technique. When access security is achieved through strong credentials, they can be periodically updated.

B.80 *Sensitivity indication* IPM

This element of service allows the originator of an IP-message to specify guidelines for the relative sensitivity of the message upon its receipt. It is the intent that the sensitivity indication should control such items as:

1) whether the recipient should have to prove his identity to receive the IP-message;
2) whether the IP-message should be allowed to be printed on a shared printer;
3) whether an IPM UA should allow the recipient to forward the received IP-message;
4) whether the IP-message should be allowed to be auto-forwarded.

The sensitivity indication can be indicated to the recipient or interpreted directly by the recipient's IPM UA.

If no sensitivity level is indicated, it should be assumed that the IP-message originator has advised no restriction on the recipient's further disposition of the IP-message. The recipient is free to forward, print, or otherwise do as he chooses with the IP-message.

Three specific levels of sensitivity above the default are defined:

- *Personal*: The IP-message is sent to the recipient as an individual, rather than to him in his role. There is no implication that the IP-message is private, however.
- *Private*: The IP-message contains information that should be seen (or heard) only by the recipient, and not by anyone else. The recipient's IPM UA can provide services to enforce this intent on behalf of the IP-message originator.
- *Company-confidential*: The IP-message contains information that should be treated according to company-specific procedures.

B.81 *Special delivery* PD PR

This element of service allows an originating user to instruct the PDS to transport the letter produced form the MHS message through the ordinary letter mail circulation system and to deliver it by special messenger delivery.

B.82 *Stored message alert* MS

This element of service allows a user of an MS to register relevant sets of criteria that can cause an alert to be generated to the user when a message arrives at the MS satisfying the selected criteria. The generation of the alert can occur as follows:

1) if the UA is connected and on-line to the MS, the alert message will be sent to the UA as soon as a message arrives at the MS that safisfies the registered criteria for generating alerts. If the UA is off line then the next time the UA connects to his MS after a message arrives at the MS satisfying the registered criteria, the user will be informed that one or more alert cases have occurred, the details of which can be determined by performing a Stored Message Summary;
2) in addition to, or as an alternative to 1) above, the MS can use other mechanisms to inform the user.

B.83 *Stored message auto-forward* MS

This element of service allows a user of an MS to register requests that the MS auto-forward selected messages that are delivered to it. The user of the MS can select through registration several sets of criteria chosen from the attributes available in the MS, and messages meeting each set of criteria will be auto-forwarded to one or more users or DLs. One text per selection criteria can also be specified to be included with each auto-forwarded message.

B.84 *Stored message deletion* MS

This element of service enables a recipient UA to delete certain of its messages from the MS. Messages cannot be deleted if they have not been previously listed.

B.85 *Stored message fetching* MS

This element of service enables a recipient UA to fetch from the MS a message, or portions of a message. The UA can fetch a message (or message portion) based on the same search criteria that can be used for stored message listing.

B.86 *Stored message listing* MS

This element of service provides a recipient UA with a list of information about certain of its messages stored in the MS. The information comprises selected attributes from a message's envelope and content and others added by the MS. The UA can limit the number of messages that will be listed.

B.87 *Stored message summary* MS

This element of service provides a recipient UA with a count of the number of messages satisfying a specified criteria based on one or more attributes of the message stored in the MS.

B.88 *Subject indication* IPM

This element of service allows the originator to indicate to the recipient(s) the subject of an IP-message being sent. The subject information is to be made available to the recipient.

B.89 *Submission time stamp indication* MT

This element of service enables the MTS to indicate to the originating UA and each recipient UA the date and time at which a message was submitted to the MTS. In the case of physical delivery, this element of service also enables the PDAU to indicate the date and time of submission on the physical message.

B.90 *Typed body* IPM

This element of service permits the nature and attributes of the body of the IP-message to be conveyed along with the body. Because the body can undergo conversion, the body type can change over time.

B.91 *Undeliverable mail with return of physical message* PD PR

This element of service enables the PDS to return the physical message without delay, with reason indicated to the originator, if it cannot be delivered to the addressee. This is the default action to be taken by the PDS.

Note – In the case of "poste restante" the return of the physical message will take place after some period of time.

B.92 *Use of distribution list* MT PR

This element of service enables an originating UA to specify a distribution list in place of all the individual recipients (users or nested LDs) mentioned therein. The MTS will add the members of the list to the recipients of the message and send it to those members. Distribution lists can be members of distribution lists, in which case the list of recipients can be successively expanded at several places in the MTS.

B.93 *User/UA capabilities registration* MT

This element of service enables a UA to indicate to its MTA, through registration, the unrestricted use of any or all of the following capabilities with respect to received messages:

1) the content type(s) of messages it is willing to have delivered to it;
2) the maximum content length of a message it is willing to have delivered to it;
3) the encoded information type(s) of messages it is willing to have delivered to it.

The MTA will not deliver to a UA a message that does not match, or exceeds, the capabilities registered.

ANNEX C

(to Recommendation X.400)

Elements of service modifications with respect to the 1984 version

C.1 *New elements of service in 1988* (see Table C-1/X.400)

TABLE C-1/X.400

Elements of service	MT	IPM	PD	MS	Annex B Ref.
Additional physical rendition			X		B.2
Basic physical rendition			X		B.7
Content confidentiality	X				B.10
Content integrity	X				B.11
Conversion prohibition in case of loss of information	X				B.14
Counter collection			X		B.16
Counter collection with advice			X		B.17
Delivery via Bureaufax service			X		B.23
Designation of recipient by directory name	X				B.24
DL expansion history indication	X				B.26
DL expansion prohibited	X				B.27
EMS (express mail service)			X		B.28
Incomplete copy indication		X			B.36
Language indication		X			B.38
Latest delivery designation	X				B.39
Message flow confidentiality	X				B.40
Message origin authentication	X				B.42
Message security labelling	X				B.43
Message sequence integrity	X				B.44
Non-repudiation of delivery	X				B.49
Non-repudiation of origin	X				B.50
Non-repudiation of submission	X				B.51
Ordinary mail			X		B.53
Originator requested alternate recipient	X				B.56
Physical delivery notification by MHS			X		B.57
Physical delivery notification by PDS			X		B.58
Physical forwarding allowed			X		B.59
Physical forwarding prohibited			X		B.60
Probe origin authentication	X				B.64
Proof of delivery	X				B.65
Proof of submission	X				B.66
Redirection d is allowed by originator	X				B.68
Redirection of incoming messages	X				B.69
Registered mail			X		B.70
Registered mail to addressee in person			X		B.71
Report origin authentication	X				B.74
Request for forwarding address			X		B.75
Requested delivery method	X				B.76
Restricted delivery	X				B.77
Secure access management	X				B.79
Special delivery			X		B.81
Stored message alert				X	B.82
Stored message auto-forward				X	B.83
Stored message deletion				X	B.84
Stored message fetching				X	B.85
Stored message listing				X	B.86
Stored message summary				X	B.87
Undeliverable mail with return of physical message			X		B.91
Use of distribution list	X				B.92
User/UA capabilities registration	X				B.93

C.2 *Mapping of the 1984 and 1988 elements of service tables* (see Figure C-1/X.400)

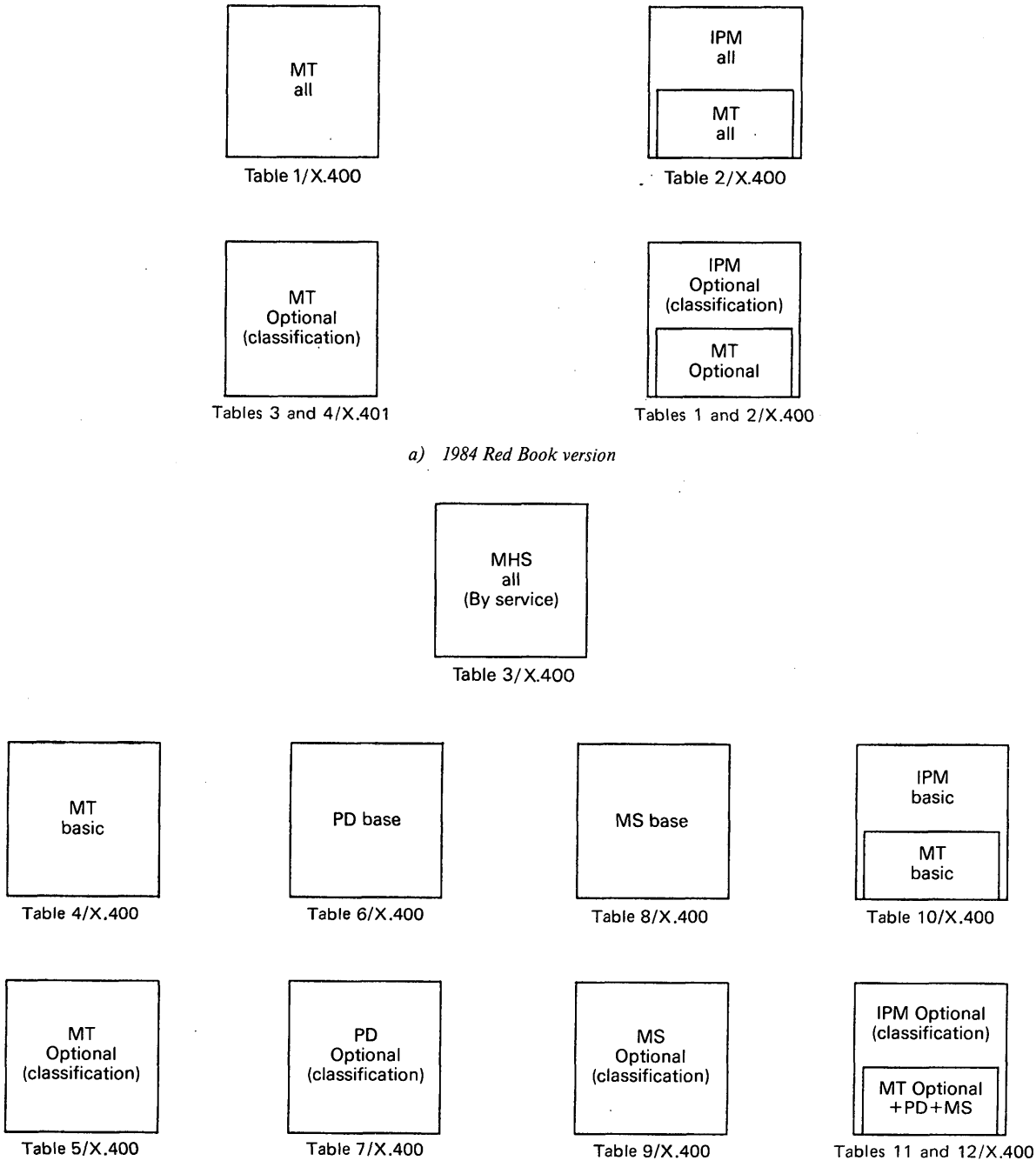


FIGURE C-1/X.400

Mapping of elements of service tables

C.3 *Classification of new elements of service*

The new elements of service that were added to the 1984 X.400-Series to create the 1988 F.400/X.400-Series Recommendations are all classified as additional optional user facilities with the following exceptions:

C.3.1 *MT service*

- DL expansion history indication;
- requested delivery method.

C.3.2 *IPM service*

- DL expansion history indication;
- language indication;
- requested delivery method.

C.3.3 *MH/PD service intercommunication*

Although some of the elements of service used in this intercommunication are classified as *Base* (see X.400, § 19.4), and some are classified as essential optional user facilities (see X.400, § 19.5), the provision of MH/PD service intercommunication is an option itself. When this intercommunication is provided, the base elements of service and optional user facilities shall be supported as classified in this Recommendation.

C.3.4 *Message store*

Although some of the elements of service used with the message store are classified as *Base* (see X.400, § 19.6), and others are classified as essential optional user facilities (see X.400, § 19.7), the provision of a message store is itself an option, and therefore the classifications are applicable only for the provider of a message store.

C.4 *Changes in classification of 1984 elements of service*

All the elements of service in 1984 have retained their 1984 classifications with the following exception:

- Non-receipt notification request.

C.4.1 *Miscellaneous changes*

The element of service registered encoded information types in 1984 is now called user/UA capabilities registration and it has been extended in functionality.

Some of the 1984 element of service definitions have been revised editorially for ease of reading.

ANNEX D

(to Recommendation X.400)

Differences between CCITT Recommendation X.400 and ISO Standard 10021-1

(This Annex is not a part of this Recommendation)

This Annex points out the major differences between this Recommendation and the corresponding ISO International Standard. Because the differences in many cases involve the inclusion or exclusion of a word, a phrase, or a sentence, and these occur in many places throughout the text, this Annex does not specifically point to these instances. Rather it summarizes the intent of these differences.

The following are the major differences:

1) The CCITT text makes references throughout to CCITT services and their relationship to MHS;

2) Figure 5/X.400 showing relationships between management domains and corresponding notes;

3) Roles of ADMD and PRMD in naming;

4) Use of MHS in provision of public services (§ 17);

5) The note about responsibility for storing deferred delivery messages (Annex B, § B.19) is not included in the ISO text.

Recommendation X.402

MESSAGE HANDLING SYSTEMS: OVERALL ARCHITECTURE[1)]

(Melbourne, 1988)

The establishment in various countries of telematic services and computer-based store-and-forward message services in association with public data networks creates a need to produce standards to facilitate international message exchange between subscribers to such services.

The CCITT,

considering

(a) the need for Message Handling Systems;

(b) the need to transfer and store messages of different types;

(c) that Recommendation X.200 defines the Reference Model of Open Systems Interconnection for CCITT applications;

(d) that Recommendations X.208, X.217, X.218 and X.219 provide the foundation for CCITT applications;

(e) that the X.500-series Recommendations define Directory Systems;

(f) that Message Handling Systems are defined in a series of Recommendations: X.400, X.402, X.403, X.407, X.408, X.411, X.413, and X.419;

(g) that Interpersonal Messaging is defined in Recommendations X.420 and T.330,

unanimously declares

(1) that the abstract models of a Message Handling System are defined in § 2;

(2) that the configurations of a Message Handling System are defined in § 3;

(3) that naming, addressing, and routing within Message Handling Systems are defined in § 4;

(4) that the use of the Directory by Message Handling Systems is defined in § 5;

(5) that the OSI realization of a Message Handling System is specified in § 6.

[1)] Recommendation X.402 and ISO 10021-2 [Information Processing Systems – Text Communications – MOTIS – Overall Architecture] were developed in close collaboration and are technically aligned, except for the differences noted in Annex F.

TABLE OF CONTENTS

SECTION 1 – INTRODUCTION

0 Introduction

This Recommendation is one of a set of Recommendations for Message Handling. The entire set provides a comprehensive blueprint for a Message Handling System (MHS) realized by any number of cooperating open systems.

The purpose of an MHS is to enable users to exchange messages on a store-and-forward basis. A message submitted on behalf of one user, the originator, is conveyed by the Message Transfer System (MTS) and subsequently delivered to the agents of one or more additional users, the recipients. Access units (AUs) link the MTS to communication systems of other kinds (e.g. postal systems). A user is assisted in the preparation, storage and display of messages by a user agent (UA). Optionally, he is assisted in the storage of messages by a message store (MS). The MTS comprises a number of message transfer agents (MTAs) which collectively perform the store-and-forward message transfer function.

The Recommendation specifies the overall architecture of the MHS and serves as a technical introduction to it.

The text of this Recommendation is the subject of joint CCITT-ISO agreement. The corresponding ISO specification is ISO 10021-2.

1 Scope

This Recommendation defines the overall architecture of the MHS and serves as a technical introduction to it.

Other aspects of Message Handling are specified in other Recommendations. A non-technical overview of Message Handling is provided by Recommendation X.400. The conformance testing of MHS components is described in Recommendation X.403. The conventions used in the definition of the abstract services provided by MHS components are defined in Recommendation X.407. The detailed rules by which the MTS converts the contents of messages from one EIT to another are defied in Recommendation X.408. The abstract service the MTS provides and the procedures that govern its distributed operation are defined in Recommendation X.411. The abstract service the MS provides is defined in Recommendation X.413. The application protocols that govern the interactions of MHS components are specified in Recommendation X.419. The Interpersonal Messaging System, an application of MHS Handling, is defined in Recommendation X.420. Telematic access to the Interpersonal Messaging System is specified in Recommendation T.330.

The CCITT Recommendations and ISO International Standards of Message Handling are summarized in Table 1/X.402.

TABLE 1/X.402

Specifications for message handling systems

CCITT	ISO	Subject matter
		Introduction
X.400 X.402	8505-1 8505-2	Service and system overview Overall architecture
		Various aspects
X.403 X.407 X.408	– 8883-2 –	Conformance testing Abstract service definition conventions Encoded information type conversion rules
		Abstract services
X.411 X.413	8883-1 TBS-1	MTS abstract service definition and procedures for distributed operation MS abstract service definition
		Protocols
X.419	8505-2	Protocol specifications
		Interpersonal messaging system
X.420 T.330	9065 –	Interpersonal messaging system Telematic access to IPMS

The Directory, the principal means for disseminating communication-related information among MHS components, is defined in the X.500-series Recommendations (see Table 2/X.402).

The architectural foundation for Message Handling is provided by other Recommendations. The OSI Reference Model is defined in Recommendation X.200. The notation for specifying the data structures of abstract services and application protocols, ASN.1, and the associated encoding rules are defined in Recommendations X.208 and X.209. The means for establishing and releasing associations, the ACSE, is defined in Recommendations X.217 and X.227. The means for reliably conveying APDUs over associations, the RTSE, is defined by Recommendations X.218 and X.228. The means for making requests of other open systems, the ROSE, is defined in Recommendations X.219 and X.229.

The CCITT Recommendations and ISO International Standards basic to Message Handling are summarized in Table 3/X.402.

TABLE 2/X.402

Specifications for directories

CCITT	ISO	Subject matter
Model		
X.200	7498	OSI reference model
X.500	9594-1	Overview
X.501	9594-2	Models
X.509	9594-8	Authentication framework
X.511	9594-3	Abstract service definition
X.518	9594-4	Procedures for distributed operation
X.519	9594-5	Protocol specifications
X.520	9594-6	Selected attribute types
X.521	9594-7	Selected object classes

TABLE 3/X.402

Specifications for MHS foundations

CCITT	ISO	Subject matter
Model		
X.200	7498	OSI reference model
ASN.1		
X.208	8824	Abstract syntax notation
X.209	8825	Basic encoding rules
Association control		
X.217	8649	Service definition
X.227	8650	Protocol specification
Reliable transfer		
X.218	9066-1	Service definition
X.228	9066-2	Protocol specification
Remote operations		
X.219	9072-1	Service definition
X.229	9072-2	Protocol specification

This Recommendation is structured as follows. Section 1 is this introduction. Section 2 presents abstract models of Message Handling. Section 3 specifies how one can configure the MHS to satisfy any of a variety of functional, physical and organizational requirements. Section 4 describes the naming and addressing of users and distribution lists and the routing of information objects to them. Section 5 describes the uses the MHS may make of the Directory. Section 6 describes how the MHS is realized by means of OSI. Annexes provide important supplemental information.

No requirements for conformance to this Recommendation are imposed.

2 References

This Recommendation and others in the set cite the documents below.

2.1 *Open systems interconnection*

This Recommendation and others in the set cite the following OSI specifications:

X.200 Reference model of open systems interconnection for CCITT applications (see also ISO 7498)

X.208 Specification of abstract syntax notation one (ASN.1) (see also ISO 8824)

X.209 Specification of basic encoding rules for abstract syntax notation (see also ISO 8825)

X.217 Association control service definition for open systems interconnection for CCITT applications (see also ISO 8649)

X.218 Reliable transfer: Model and service definition (see also ISO 9066-1)

X.219 Remote operations: Model, notation and service definition (see also ISO 9072-1)

X.227 Association control: Protocol specification for open systems interconnection for CCITT applications (see also ISO 8650)

X.228 Reliable transfer: Protocol specification (see also ISO 9066-2)

X.229 Remote operations: Protocol specification (see also ISO 9072-2)

2.2 *Directory systems*

This Recommendation and others in the set cite the following Directory System specifications:

X.500 The directory – Overview of concepts, models, and services (see also ISO 9594-1)

X.501 The directory – Models (see also ISO 9594-2)

X.509 The directory – Authentication framework (see also ISO 9594-8)

X.511 The directory – Abstract service definition (see also ISO 9594-3)

X.518 The directory – Procedures for distributed operation (see also ISO 9594-4)

X.519 The directory – Protocol specifications (see also ISO 9594-5)

X.520 The directory – Selected attribute types (see also ISO 9594-6)

X.521 The directory – Selected object classes (see also ISO 9594-7)

2.3 *Message handling systems*

This Recommendation and others in the set cite the following message handling system specifications:

T.330 Telematic access to interpersonal messaging system

X.400 Message handling: System and service overview (see also ISO 10021-1)

X.403 Message handling systems: Conformance testing

X.407 Message handling systems: Abstract service definition conventions (see also ISO 10021-3)

X.408 Message handling systems: Encoded information type conversion rules

X.411 Message handling systems: Message transfer system: Abstract service definition and procedures (see also ISO 10021-4)

X.413 Message handling systems: Message store: Abstract service definition (see also ISO 10021-5)

X.419 Message handling systems: Protocol specifications (see also ISO 10021-6)

X.420 Message handling systems: Interpersonal messaging system (see also ISO 10021-7)

3 Definitions

For the purposes of this Recommendation and others in the set, the definitions below apply.

3.1 *Open systems interconnection*

3.1.1 This Recommendation and others in the set use the following terms defined in Recommendation X.200, as well as the names of the seven layers of the Reference Model:

a) abstract syntax;
b) application entity (AE);
c) application process;
d) application protocol data unit (APDU);
e) application service element (ASE);
f) distributed information processing task;
g) layer;
h) open system;
i) open systems interconnection (OSI);
j) peer;
k) presentation context;
l) protocol;
m) reference model;
n) transfer syntax;
o) user element (UE).

3.1.2 This Recommendation and others in the set use the following terms defined in Recommendations X.208 and X.209, as well as the names of ASN.1 data types and values:

a) Abstract Syntax Notation One (ASN.1);
b) Basic Encoding Rules;
c) explicit;
d) export;
e) implicit;
f) import;
g) macro;
h) module;
i) tag;
j) type; and
k) value.

3.1.3 This Recommendation and others in the set use the following terms defined in Recommendation X.217:

a) application association; association;
b) application context (AC);
c) association control service element (ACSE);
d) initiator;
e) responder.

3.1.4 This Recommendation and others in the set use the following terms defined in Recommendation X.218:

a) Reliable transfer (RT); and
b) Reliable transfer service element (RTSE).

3.1.5 This Recommendation and others in the set use the following terms defined in Recommendation X.219:

a) argument;
b) asynchronous;
c) bind;
d) parameter;
e) remote error;
f) remote operation;
g) remote operations (RO);
h) remote operations service element (ROSE);
i) result;
j) synchronous; and
k) unbind.

3.2 *Directory systems*

This Recommendation and others in the set use the following terms defined in the X.500-series Recommendations:

a) attribute;
b) certificate;
c) certification authority;
d) certification path;
e) directory entry; entry;
f) directory system agent (DSA);
g) directory;
h) hash function;
i) name;
j) object class;
k) object;
l) simple authentication;
m) strong authentication.

3.3 *Message handling systems*

For the purposes of this Recommendation and others in the set, the definitions indexed in Annex G apply.

4 Abbreviations

For the purposes of this Recommendation and others in the set, the abbreviations indexed in Annex G apply.

5 Conventions

This Recommendation uses the descriptive conventions identified below.

5.1 *ASN.1*

This Recommendation uses several ASN.1-based descriptive conventions in Annexes A and C to define the Message Handling-specific information the Directory may hold. In particular, it uses the OBJECT-CLASS, ATTRIBUTE, and ATTRIBUTE-SYNTAX macros of Recommendation X.501 to define Message Handling-specific object classes, attributes, and attribute syntaxes.

ASN.1 appears both in Annex A to aid the exposition, and again, largely redundantly, in Annex C for reference. If differences are found between the two, a specification error is indicated.

Note that ASN.1 tags are implicit throughout the ASN.1 module that Annex C defines; the module is definitive in that respect.

5.2 *Grade*

Whenever this Recommendation describes a class of data structure (e.g., O/R addresses) having components (e.g., attributes), each component is assigned one of the following **grades**:

a) **mandatory (M)**: a mandatory component shall be present in every instance of the class.

b) **optional (O)**: an optional component shall be present in an instance of the class at the discretion of the object (e.g., user) supplying that instance. There is no default value.

c) **defaultable (D)**: a defaultable component shall be present in an instance of the class at the discretion of the object (e.g., user) supplying that instance. In its absence a default value, specified by this Recommendation, applies.

d) **conditional (C)**: a conditional component shall be present in an instance of the class as dictated by this Recommendation.

5.3 *Terms*

Throughout the remainder of this Recommendation, terms are rendered in **bold** when defined, in *italic* when referenced prior to their definitions, without emphasis upon other occasions.

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

SECTION 2 – ABSTRACT MODELS

6 Overview

This section presents abstract models of *Message Handling* which provide the architectural basis for the more detailed specifications that appear in other Recommendations in the set.

Message Handling is a distributed information processing task that integrates the following intrinsically related sub-tasks:

a) **Message Transfer**: The non-real-time carriage of information objects between parties using computers as intermediaries.

b) **Message Storage**: The automatic storage for later retrieval of information objects conveyed by means of Message Transfer.

This section covers the following topics:

a) functional model;

b) information model;

c) operational model;

d) security model.

Note – Message Handling has a variety of applications, one of which is Interpersonal Messaging, described in Recommendation X.420.

7 Functional model

This clause provides a functional model of Message Handling. The concrete realization of the model is the subject of other Recommendations in the set.

The **Message Handling Environment**; comprises "primary" functional objects of several types, the *Message Handling System (MHS), users,* and *distribution lists.* The MHS in turn can be decomposed into lesser, "secondary" functional objects of several types, the *Message Transfer System (MTS), user agents, message stores,* and *access units.* The MTS in turn can be decomposed into still lesser, "tertiary" functional objects of a single type, message transfer agents.

The primary, secondary, and tertiary functional object types and selected *access unit* types are individually defined and described below.

As detailed below, functional objects are sometimes tailored to one or more applications of Message Handling, e.g., Interpersonal Messaging (see Recommendations X.420 and T.330). A functional object that has been tailored to an application understands the syntax and semantics of the contents of messages exchanged in that application.

As a local matter, functional objects may have capabilities beyond those specified in this Recommendation or others in the set. In particular, a typical *user agent* has message preparation, rendition, and storage capabilities that are not standardized.

7.1 *Primary functional objects*

The MHE comprises the *Message Handling System*, *users*, and *distribution lists*. These primary functional objects interact with one another. Their types are defined and described below.

The situation is depicted in Figure 1/X.402.

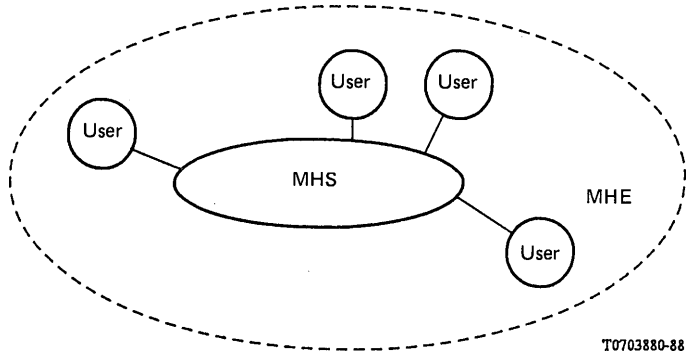


FIGURE 1/X.402

Message handling environment

7.1.1 *Message Handling System*

The principal purpose of Message Handling is to convey information objects from one party to another. The functional object by means of which this is accomplished is called the **Message Handling System.**

The MHE comprises a single MHS.

7.1.2 *Users*

The principal purpose of the MHS is to convey information objects between *users*. A functional object (e.g., a person) that engages in (rather than provides) Message Handling is called a **user.**

The following kinds of user are distinguished:

a) **direct user**: A user that engages in Message Handling by direct use of the MHS.

b) **indirect user**: A user that engages in Message Handling by indirect use of the MHS, i.e., through another communication system (e.g., a postal system or the telex network) to which the MHS is linked.

The MHE comprises any number of users.

7.1.3 *Distribution lists*

By means of the MHS a user can convey information objects to pre-specified groups of users as well as to individual users. The functional object that represents a pre-specified group of users and other DLs is called a **distribution list (DL).**

A DL identifies zero or more users and DLs called its **members**. The latter DLs (if any) are said to be nested. Asking the MHS to convey an information object (e.g., a *message*) to a DL is tantamount to asking that it convey the object to its members. Note that this is recursive.

The right, or permission, to convey *messages* to a particular DL may be controlled. This right is called **submit permission**. As a local matter the use of a DL can be further restricted.

The MHE comprises any number of DLs.

Note – A DL might be further restricted, e.g., to the conveyance of *messages* of a prescribed *content type.*

7.2 *Secondary functional objects*

The MHS comprises the *Message Transfer System, user agents, message stores,* and *access units.* These secondary functional objects interact with one another. Their types are defined and described below.

The situation is depicted in Figure 2/X.402.

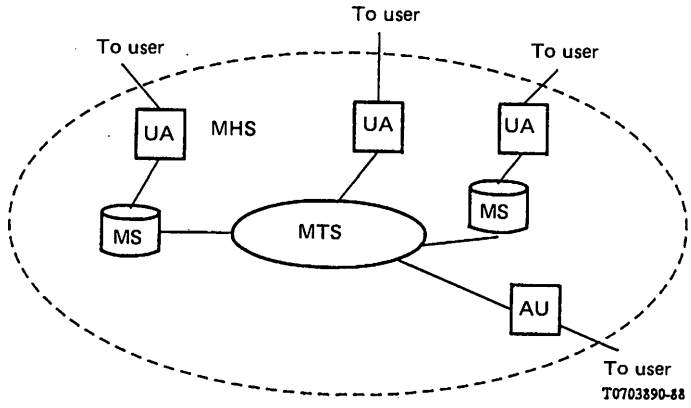


FIGURE 2/X.402

Message handling system

7.2.1 *Message Transfer System*

The MHS conveys information objects to individual users and to the members of DLs. The functional object that actually does this is called the **Message Transfer System (MTS).** The MTS is a store-and-forward communication system and can be considered the backbone of the MHS.

The MTS is general-purpose, supporting all applications of Message Handling. Additionally, the MTS may be tailored to one or more particular applications so it can carry out *conversion.*

The MHS comprises a single MTS.

7.2.2 *User agents*

The functional object by means of which a single direct user engages in Message Handling is called a **user agent (UA).**

A typical UA is tailored to one or more particular applications of Message Handling.

The MHS comprises any number of UAs.

Note – A UA that serves a human user typically interacts with him by means of input/output devices (e.g., a keyboard, display, scanner, printer, or combination of these).

7.2.3 *Message stores*

A typical user must store the information objects it receives. The functional object that provides a (single) direct user with capabilities for Message Storage is called a **message store (MS).** Each MS is associated with one UA, but not every UA has an associated MS.

Every MS is general-purpose, supporting all applications of Message Handling. Additionally, an MS may be tailored to one or more particular applications so that it can more capably *submit* and support the *retrieval of messages* associated with that application.

The MHS comprises any number of MSs.

Note – As a local matter a UA may provide for information objects storage that either supplements or replaces that of an MS.

7.2.4 *Access units*

The functional object that links another communication system (e.g., a postal system or the telex network) to the MTS and via which its patrons engage in Message Handling as indirect users is called an **access unit (AU).**

A typical AU is tailored to a particular communication system and to one or more particular applications of Message Handling.

The MHS comprises any number of AUs.

7.3 *Tertiary functional objects*

The MTS comprises *message transfer agents.* These tertiary functional objects interact. Their type is defined and described below.

The situation is depicted in Figure 3/X.402.

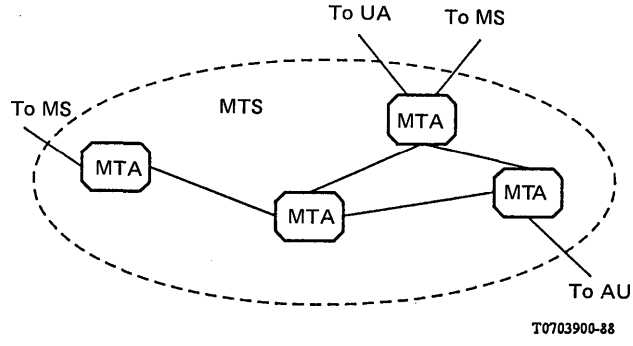


FIGURE 3/X.402

Message transfer system

7.3.1 *Message transfer agents*

The MTS conveys information objects to users and DLs in a store-and-forward manner. A functional object that provides one link in the MTS' store-and-forward chain is called a **message transfer agent (MTA).**

Every MTA is general-purpose, supporting all applications of Message Handling. Additionally, an MTA may be tailored to one or more particular applications so it can carry out *conversion.*

The MTS comprises any number of MTAs.

7.4 *Selected AU types*

As described above, the MHS interworks with communication systems of other types via AUs. Several selected AU types – *physical delivery,* telematic, and telex – are introduced in the clauses below.

7.4.1 *Physical delivery*

A **physical delivery access unit (PDAU)** is an AU that subjects *messages* (but neither *probes* nor *reports*) to *physical rendition* and that conveys the resulting *physical messages* to a *physical delivery system.*

The transformation of a *message* into a *physical message* is called **physical rendition**. A **physical message** is a physical object (e.g., a letter and its paper envelope) that embodies a *message*.

A **physical delivery system (PDS)** is a system that performs *physical delivery*. One important kind of PDS is postal systems. **Physical delivery** is the conveyance of a physical message to a patron of a PDS, one of the indirect users to which the PDAU provides Message Handling capabilities.

Among the applications of Message Handling supported by every PDAU is Interpersonal Messaging (see Recommendation X.420).

7.4.2 *Telematic*

Telematic access units, which support Interpersonal Messaging exclusively, are introduced in Recommendation X.420.

7.4.3 *Telex*

Telex access units, which support Interpersonal Messaging exclusively, are introduced in Recommendation X.420.

8 Information model

This clause provides an information model of Message Handling. The concrete realization of the model is the subject of other Recommendations in the set.

The MHS and MTS can convey information objects of three classes: *messages*, *probes* and *reports*. These classes are listed in the first column of Table 4/X.402. For each listed class, the second column indicates the kinds of functional objects – users, UAs, MSs, MTAs, and AUs – that are the ultimate sources and destinations for such objects.

TABLE 4/X.402

Conveyable information objects

Information object	Functional object				
	User	UA	MS	MTA	AU
Message	SD	–	–	–	–
Probe	S	–	–	D	–
Report	D	–	–	S	–

S Ultimate source

D Ultimate destination

The information objects, summarized in Table 4/X.402, are individually defined and described below.

8.1 *Messages*

The primary purpose of Message Transfer is to convey information objects called **messages** from one user to others. A message has the following parts, as depicted in Figure 4/X.402:

a) **envelope**: An information object whose composition varies from one *transmittal step* to another and that variously identifies the message's *originator* and *potential recipients*, documents its previous conveyance and directs its subsequent conveyance by the MTS, and characterizes its *content*.

b) **content**: An information object that the MTS neither examines nor modifies, except for *conversion*, during its conveyance of the message.

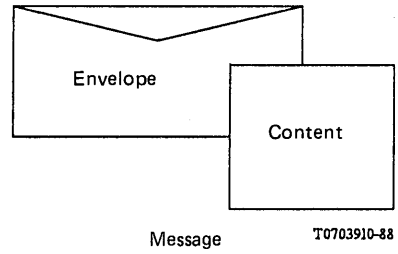


FIGURE 4/X.402

A message's envelope and content

One piece of information borne by the envelope identifies the type of the content. The **content type** is an identifier (an ASN.1 Object Identifier or Integer) that denotes the syntax and semantics of the content overall. This identifier enables the MTS to determine the message's *deliverability* to particular users, and enables UAs and MSs to interpret and process the content.

Another piece of information borne by the envelope identifies the types of encoded information represented in the content. An **encoded information type (EIT)** is an identifier (an ASN.1 Object Identifier or Integer) that denotes the medium and format (e.g., IA5 text or Group 3 facsimile) of individual portions of the content. It further enables the MTS to determine the message's deliverability to particular users, and to identify opportunities for it to *make* the message deliverable by converting a portion of the content from one EIT to another.

8.2 *Probes*

A second purpose of Message Transfer is to convey information objects called **probes** from one user up to but just short of other users (i.e., to the MTAs serving those users). A probe describes a class of message and is used to determine the *deliverability* of such messages.

A message described by a probe is called a **described message**.

A probe comprises an envelope alone. This envelope contains much the same information as that for a message. Besides bearing the content type and encoded information types of a described message, the probe's envelope bears the length of its content.

The *submission* of a probe elicits from the MTS largely the same behavior as would submission of any described message, except that *DL expansion* and *delivery* are forgone in the case of the probe. In particular, and apart from the consequences of the suppression of *DL expansion*, the probe provokes the same *reports* as would any described message. This fact gives probes their utility.

8.3 *Reports*

A third purpose of Message Transfer is to convey information objects called **reports** to users. Generated by the MTS, a report relates the outcome or progress of a message's or probe's *transmittal* to one or more potential recipients.

The message or probe that is the subject of a report is called its **subject message** or **subject probe**.

A report concerning a particular potential recipient is conveyed to the *originator* of the subject message or probe unless the *potential recipient* is a *member recipient*. In the latter case, the report is conveyed to the DL of which the *member recipient* is a member. As a local matter (i.e., by policy established for that particular DL), the report may be further conveyed to the DL's owner; either to another, containing DL (in the case of nesting) or to the originator of the subject message (otherwise); or both.

The outcomes that a single report may relate are of the following kinds:

a) **delivery report** *delivery, export,* or *affirmation* of the subject message or probe, or *DL expansion.*

b) **non-delivery report** *non-delivery* or *non-affirmation* of the subject message or probe.

A report may comprise one or more delivery and/or non-delivery reports.

A message or probe may provoke several delivery and/or non-delivery reports concerning a particular *potential recipient.* Each marks the passage of a different transmittal *step* or *event.*

9 Operational model

This clause provides an operational model of Message Handling. The concrete realization of the model is the subject of other Recommendations in the set.

The MHS can convey an information object to individual users, DLs, or a mix of the two. Such conveyance is accomplished by a process called *transmittal* comprising *steps* and *events.* The process, its parts, and the roles that users and DLs play in it are defined and described below.

9.1 *Transmittal*

The conveyance or attempted conveyance of a message or probe is called **transmittal.** Transmittal encompasses a message's conveyance from its *originator* to its *potential recipients,* and a probe's conveyance from its *originator* to MTAs able to *affirm* the described messages' *deliverability* to the probe's *potential recipients.* Transmittal also encompasses the conveyance or attempted conveyance to the *originator* of any reports the message or probe may provoke.

A transmittal comprises a sequence of *transmittal steps* and *events.* A **transmittal step** (or **step**) is the conveyance of a message, probe, or report from one functional object to another "adjacent" to it. A **transmittal event** (or **event**) is processing of a message, probe, or report within a functional object that may influence the functional object's selection of the next transmittal step or event.

The information flow of transmittal is depicted in Figure 5/X.402. The figure shows the kinds of functional objects – direct users, indirect users, UAs, MSs, MTAs, and AUs – that may be involved in a transmittal, the information objects – messages, probes, and reports – that may be conveyed between them, and the names of the transmittal steps by means of which those conveyances are accomplished.

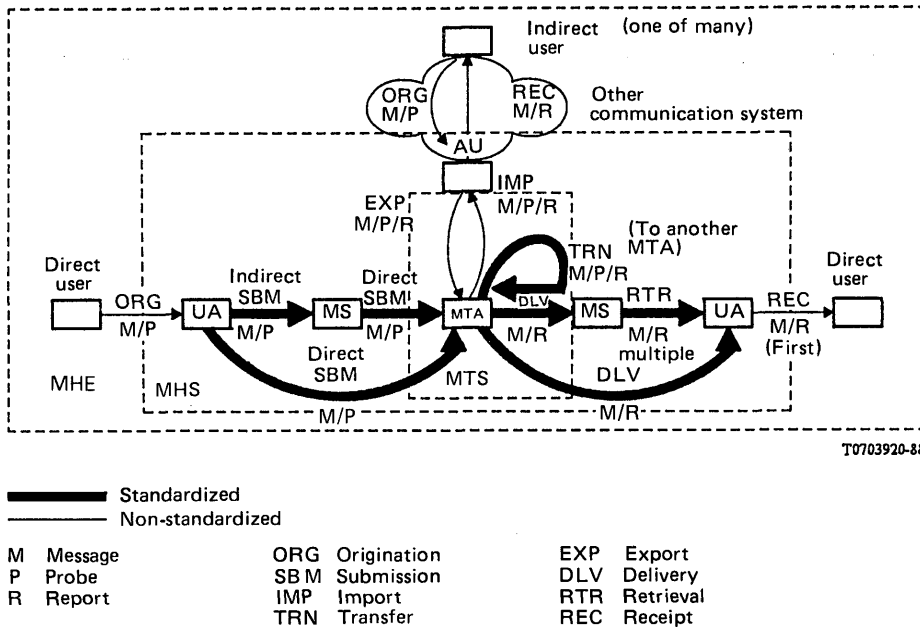


FIGURE 5/X.402

The Information flow of transmittal

The Figure highlights the facts that a message or report may be retrieved repeatedly and that only the first conveyance of a retrieved object from UA to user constitutes *receipt.*

One event plays a distinguished role in transmittal. *Splitting* replicates a message or probe and divides responsibility for its *immediate recipients* among the resulting information objects. The potential recipients associated with a particular instance of a message or probe are called the **immediate recipients**. An MTA stages a splitting if the next step or event required in the conveyance of a message or probe to some immediate recipients differs from that required in its conveyance to others. Each of the step and event descriptions which follow assumes that the step or event is appropriate for all immediate recipients, a situation that can be created, if necessary, by splitting.

9.2 *Transmittal roles*

Users and DLs play a variety of roles in a message's or probe's transmittal. These roles are informally categorized as "source" roles, "destination" roles, or statuses to which users or DLs can be elevated.

9.2.1 A user may play the following "source" role in the transmittal of a message or probe:

a) **originator**: The user (but not DL) that is the ultimate source of a message or probe.

9.2.2 A user or DL may play any of the following "destination" roles in the transmittal of a message or probe:

a) **intended recipient**: One of the users and DLs the originator specifies as a message's or probe's intended destinations.

b) **originator-specified alternate recipient**: The user or DL (if any) to which the originator requests that a message or probe be conveyed if it cannot be conveyed to a particular intended recipient.

c) **member recipient**: A user or DL to which a message (but not a probe) is conveyed as a result of DL *expansion.*

d) **recipient-assigned alternate recipient**: The user or DL (if any) to which an intended, originator-specified alternate, or member recipient may have elected to *redirect* messages.

9.2.3 A user or DL may attain any of the following statuses in the course of a message's or probe's transmittal:

a) **potential recipient**: Any user or DL to (i.e., toward) which a message or probe is conveyed at any point during the course of transmittal. Necessarily an intended, originator-specified alternate, member, or recipient-assigned alternate recipient.

b) **actual recipient** (or **recipient**): A potential recipient for which *delivery* or *affirmation* takes place.

9.3 *Transmittal steps*

The kinds of steps that may occur in a transmittal are listed in the first column of Table 5/X.402. For each listed kind, the second column indicates whether this Recommendation and others in the set standardize such steps, the third column the kinds of information objects – messages, probes, and reports – that may be conveyed in such a step, the fourth column the kinds of functional objects – users, UAs, MSs, MTAs, and AUs – that may participate in such a step as the object's source or destination.

The Table is divided into three sections. The steps in the first section apply to the "creation" of messages and probes, those in the last to the "disposal" of messages and reports, and those in the middle section to the "relaying" of messages, probes, and reports.

The kinds of transmittal steps, summarized in Table 5/X.402, are individually defined and described below.

TABLE 5/X.402

Transmittal steps

Transmittal step	Standardized?	Information objects			Functional objects				
		M	P	R	User	UA	MS	MTA	AU
Origination	No	X	X	–	S	D	–	–	–
Submission	Yes	X	X	–	–	S	SD	D	–
Import	No	X	X	X	–	–	–	D	S
Transfer	Yes	X	X	X	–	–	–	SD	–
Export	No	X	X	X	–	–	–	S	D
Delivery	Yes	X	–	X	–	D	D	S	–
Retrieval	Yes	X	–	X	–	D	S	–	–
Reception	No	X	–	X	D	S	–	–	–

M Message S Source

P Probe D Destination

R Report X Permitted

9.3.1 *Origination*

In an **origination** step, either a direct user conveys a message or probe to its UA, or an indirect user conveys a message or probe to the communication system that serves it. This step gives birth to the message or probe and is the first step in its transmittal.

The user above constitutes the message's or probe's originator. In this step, the originator identifies the message's or probe's intended recipients. Additionally, for each intended recipient, the originator may (but need not) identify an originator-specified alternate recipient.

9.3.2 *Submission*

In a **submission** step, a message or probe is conveyed to an MTA and thus entrusted to the MTS. Two kinds of submission are distinguished:

a) **indirect submission**: A transmittal step in which the originator's UA conveys a message or probe to its MS and in which the MS effects direct submission. Such a step follows origination.

   This step may be taken only if the user is equipped with an MS.

b) **direct submission**: A transmittal step in which the originator's UA or MS conveys a message or probe to an MTA. Such a step follows origination or occurs as part of indirect submission.

   This step may be taken whether or not the user is equipped with an MS.

Indirect and direct submission are functionally equivalent except that additional capabilities may be available with the former. Indirect submission may differ from direct submission in other respects (e.g., the number of open systems with which that embodying a UA must interact) and for that reason be preferable to direct submission.

The UA or MS involved in direct submission is called the **submission agent**. A submission agent is made known to the MTS by a process of registration, as a result of which the submission agent and MTS keep one another informed of their names, their locations, and any other characteristics required for their interaction.

9.3.3 *Import*

In an **import** step, an AU conveys a message, probe, or report to an MTA. This step injects into the MTS an information object born in another communication system, and follows its conveyance by that system.

Note – The concept of importing is a generic one. How this step is effected varies, of course, from one type of AU to another.

9.3.4 *Transfer*

In a **transfer** step, one MTA conveys a message, probe, or report to another. This step transports an information object over physical and sometimes organizational distances and follows direct submission, import, or (a prior) transfer.

This step may be taken, of course, only if the MTS comprises several MTAs.

The following kinds of transfer are distinguished, on the basis of the number of MDs involved:

a) **internal transfer**: A transfer involving MTAs within a single *MD*.

b) **external transfer**: A transfer involving MTAs in different *MD*s.

9.3.5 *Export*

In an **export** step, an MTA conveys a message, probe, or report to an AU. This step ejects from the MTS an information object bound for another communication system. It follows direct submission, import, or transfer.

As part of this step, the MTA may generate a delivery report.

Note – The concept of exporting is a generic one. How this step is effected varies, of course, from one type of AU to another.

9.3.6 *Delivery*

In a **delivery** step, an MTA conveys a message or report to an MS or UA. The MS and UA are those of a potential recipient of the message, or the originator of the report's subject message or probe. This step entrusts the information object to a representative of the user and follows direct submission, import, or transfer. It also elevates the user in question to the status of an actual recipient.

As part of this step, in the case of a message, the MTA may generate a delivery report.

The MS or UA involved is called the **delivery agent**. A delivery agent is made known to the MTS by a process of registration, as a result of which the delivery agent and MTS keep one another informed of their names, their locations, and any other characteristics required for their interaction.

9.3.7 *Retrieval*

In a **retrieval** step, a user's MS conveys a message or report to its UA. The user in question is an actual recipient of the message or the originator of the subject message or probe. This step non-destructively retrieves the information object from storage. This step follows delivery or (a prior) retrieval.

This step may be taken only if the user is equipped with an MS.

9.3.8 *Receipt*

In a **receipt** step, either a UA conveys a message or report to its direct user, or the communication system that serves an indirect user conveys such an information object to that user. In either case, this step conveys the object to its ultimate destination.

In the case of a direct user, this step follows the object's delivery or first retrieval (only). In the case of an indirect user, it follows the information object's conveyance by the communication system serving the user. In either case, the user is a potential recipient (and, in the case of a direct user, an actual recipient) of the message in question, or the originator of the subject message or probe.

9.4 *Transmittal events*

The kinds of events that may occur in a transmittal are listed in the first column of Table 6/X.402. For each listed kind, the second column indicates the kinds of information objects – messages, probes, and reports – for which such events may be staged, the third column the kinds of functional objects – users, UAs, MSs, MTAs, and AUs – that may stage such events.

All the events occur within the MTS.

TABLE 6/X.402

Transmittal events

Transmittal event	Functional objets			Functional objects				
	M	P	R	User	UA	MS	MTA	AU
Splitting	X	X	–	–	–	–	X	–
Joining	X	X	X	–	–	–	X	–
Name resolution	X	X	–	–	–	–	X	–
DL expansion	X	–	–	–	–	–	X	–
Redirection	X	X	–	–	–	–	X	–
Conversion	X	X	–	–	–	–	X	–
Non-delivery	X	–	X	–	–	–	X	–
Non-affirmation	–	X	–	–	–	–	X	–
Affirmation	–	X	–	–	–	–	X	–
Routing	X	X	X	–	–	–	X	–

M Message

P Probe

R Report

X Permitted

The kinds of transmittal events, summarized in Table 6/X.402, are individually defined and described below.

9.4.1 *Splitting*

In a **splitting** event, an MTA replicates a message or probe, dividing responsibility for its immediate recipients among the resulting information objects. This event effectively allows an MTA to independently convey an object to various potential recipients.

An MTA stages a splitting when the next step or event required in the conveyance of a message or probe to some immediate recipients differs from that required in its conveyance to others.

9.4.2 *Joining*

In a **joining** event, an MTA combines several instances of the same message or probe, or two or more delivery and/or non-delivery reports for the same subject message or probe.

An MTA may, but need not stage a joining when it determines that the same events and next step are required to convey several highly related information objects to their destinations.

9.4.3 *Name resolution*

In a **name resolution** event, an MTA adds the corresponding *O/R address* to the *O/R name* that identifies one of a message's or probe's immediate recipients.

9.4.4 *DL expansion*

In a **DL expansion** event, an MTA resolves a DL among a message's (but not a probe's) immediate recipients to its members which are thereby made member recipients. This event removes indirection from the immediate recipients' specification.

A particular DL is always subjected to DL expansion at a pre-established location within the MTS. This location is called the DL's **expansion point** and is identified by an *O/R address.*

As part of this event, the MTA may generate a delivery report.

DL expansion is subject to submit permission. In the case of a nested DL, that permission must have been granted to the DL of which the nested DL is a member. Otherwise, it must have been granted to the originator.

9.4.5 *Redirection*

In a **redirection** event, an MTA replaces a user or DL among a message's or probe's immediate recipients with an originator-specified or recipient-assigned alternate recipient.

9.4.6 *Conversion*

In a **conversion** event, an MTA transforms parts of a message's content from one EIT to another, or alters a probe so it appears that the described messages were so modified. This event increases the likelihood that an information object can be delivered or affirmed by tailoring it to its immediate recipients.

The following kinds of conversion are distinguished, on the basis of how the EIT of the information to be converted and the EIT to result from the conversion are selected:

a) **explicit conversion**: A conversion in which the originator selects both the initial and final EITs.

b) **implicit conversion**: A conversion in which the MTA selects the final EITs based upon the initial EITs and the capabilities of the UA.

9.4.7 *Non-delivery*

In a **non-delivery** event, an MTA determines that the MTS cannot deliver a message to its immediate recipients, or cannot deliver a report to the originator of its subject message or probe. This event halts the conveyance of an object the MTS deems unconveyable.

As part of this event, in the case of a message, the MTA generates a non-delivery report.

An MTA stages a non-delivery, e.g., when it determines that the immediate recipients are improperly specified, that they do not accept delivery of messages like that at hand, or that the message has not been delivered to them within pre-specified time limits.

9.4.8 *Non-affirmation*

In a **non-affirmation** event, an MTA determines that the MTS could not deliver a described message to a probe's immediate recipients. This event partially or fully determines the answer to the question posed by a probe.

As part of this event, the MTA generates a non-delivery report.

An MTA stages a non-affirmation, e.g., when it determines that the immediate recipients are improperly specified or would not accept delivery of a described message.

9.4.9 *Affirmation*

In an **affirmation** event, an MTA determines that the MTS could deliver any described message to a probe's immediate recipients. This event partially or fully determines the answer to the question posed by a probe, and elevates the immediate recipients to the status of actual recipients.

As part of this event, the MTA may generate a delivery report.

An MTA stages an affirmation once it determines that the immediate recipients are properly specified and, if the immediate recipients are users (but not DLs), would accept delivery of any described message. If the immediate recipients are DLs, and MTA stages an affirmation if the DL exists and the originator has the relevant submit permission.

9.4.10 *Routing*

In a **routing** event, an MTA selects the "adjacent" MTA to which it will transfer a message, probe, or report. This event incrementally determines an information object's route through the MTS and (obviously) may be taken only if the MTS comprises several MTAs.

The following kinds of routing are distinguished, on the basis of the kind of transfer for which they prepare:

a) **internal routing**: A routing preparatory to an internal transfer (i.e., a transfer within an MD).

b) **external routing**: A routing preparatory to an external transfer (i.e., a transfer between MDs).

An MTA stages a routing when it determines that it can stage no other event, and take no step, regarding an object.

10 Security model

This clause provides an abstract security model for Message Transfer. The concrete realization of the model is the subject of other Recommendations in the set. The security model provides a framework for describing the security services that counter potential threats (see Annex D) to the MTS and the security elements that support those services.

The security features are an optional extension to the MHS that can be used to minimise the risk of exposure of assets and resources to violations of a security policy (threats). Their aim is to provide features independently of the communications services provided by other lower or higher entities. Threats may be countered by the use of physical security, computer security (COMPUSEC), or security services provided by the MHS. Depending on the perceived threats, certain of the MHS security services will be selected in combination with appropriate physical security and COMPUSEC measures. The security services supported by the MHS are described below. The naming and structuring of the services are based on ISO 7498-2.

Note – Despite these security features, certain attacks may by be mounted against communication between a user and the MHS or against user-to-user communication (e.g. in the case of users accessing their UAs). To counter these attacks requires extensions to the present security model and services which are for further study.

In many cases, the broad classes of threats are covered by several of the services listed.

The security services are supported through use of service elements of the Message Transfer Service message envelope. The envelope contains security relevant arguments as described in Recommendation X.411. The description of the security services takes the following general form. In § 10.2 the services are listed, with, in each case, a definition of the service and an indication of how it may be provided using the security elements in Recommendation X.411. In § 10.3 the security elements are individually described, with, in each case, a definition of the service element and references to its constituent arguments in Recommendation X.411.

Many of the techniques employed rely on encryption mechanisms. The security services in the MHS allow for flexibility in the choice of algorithms. However, in some cases only the use of asymmetric encryption has been fully defined in this Recommendation. A future version of this Recommendation may make use of alternative mechanisms based on symmetric encipherment.

Note – The use of the terms "security service" and "security element" in this clause are not to be confused with the terms "service" and "element of service" as used in Recommendation X.400. The former terms are used in the present clause to maintain consistency with ISO 7498-2.

10.1 *Security policies*

Security services in the MHS must be capable of supporting a wide range of security policies which extend beyond the confines of the MHS itself. The services selected and the threats addressed will depend on the individual application and levels of trust in parts of the system.

A security policy defines how the risk to and exposure of assets can be reduced to an acceptable level.

In addition, operation between different domains, each with their own security policy, will be required. As each domain will be subject to its own overall security policy, covering more than just the MHS, a bilateral agreement on interworking between two domains will be required. This must be defined so as not to conflict with the security policies for either domain and effectively becomes part of the overall security policy for each domain.

10.2 *Security services*

This defines the Message Transfer security services. The naming and structuring of the services are based on ISO 7498-2.

Message Transfer security services fall into several broad classes. These classes and the services in each are listed in Table 7/X.402.

Throughout the security service definitions that follow, reference is made to Figure 6/X.402, which reiterates the MHS functional model in simplified form. The numeric labels are referenced in the text.

10.2.1 *Origin Authentication security services*

These security services provide for the authentication of the identity of communicating peer entities and sources of data.

10.2.1.1 *Data Origin Authentication security services*

These security services provide corroboration of the origin of a message, probe, or report to all concerned entities (i.e., MTAs or recipient MTS-users). These security services cannot protect against duplication of messages, probes, or reports.

10.2.1.1.1 *Message Origin Authentication security service*

The Message Origin Authentication service enables the corroboration of the source of a message.

This security service can be provided using either the Message Origin Authentication or the Message Argument Integrity security element. The former can be used to provide the security service to any of the parties concerned (1-5 inclusive in Figure 6/X.402), whereas the latter can only be used to provide the security service to MTS-users (1 or 5 in Figure 6/X.402). The security element chosen depends on the prevailing security policy.

10.2.1.1.2 *Probe Origin Authentication security service*

The Probe Origin Authentication security service enables the corroboration of the source of a probe.

This security service can be provided by using the Probe Origin Authentication security element. This security element can be used to provide the security service to any of the MTAs through which the probe is transferred (2-4 inclusive in Figure 6/X.402).

10.2.1.1.3 *Report Origin Authentication security service*

The Report Origin Authentication security service enables the corroboration of the source of a report.

This security service can be provided by using the Report Origin Authentication security element. This security element can be used to provide the security service to the originator of the subject message or probe, as well as to any MTA through which the report is transferred (1-5 inclusive in Figure 6/X.402).

TABLE 7/X.402

Message transfer security services

	Service							
	UA/ UA	MS/ MTA	MTA/ MS	MTA/ UA	UA/ MS	UA/ MTA	MTA/ MTA	MS/ UA
Origin authentication								
Message origin authentication	*	*	–	*	–	–	–	–
Probe origin authentication	–	–	*	*	–	–	–	–
Report origin authentication	–	–	–	–	*	*	*	–
Proof of submission	–	–	–	–	–	–	*	–
Proof of delivery	*	–	–	–	–	–	–	a)
Secure access management								
Peer entity authentication	–	*	*	*	*	*	*	*
Security context	–	*	*	*	*	*	*	*
Data confidentiality								
Connection confidentiality	–	*	*	*	*	*	*	*
Connection confidentiality	*	–	–	–	–	–	–	–
Message flow confidentiality	*	–	–	–	–	–	–	–
Data integrity services								
Connection integrity	–	*	*	*	*	*	*	*
Content integrity	*	–	–	–	–	–	–	–
Message sequence integrity	*	–	–	–	–	–	–	–
Non-repudiation								
Non-repudiation of origin	*	–	–	*	–	–	–	–
Non-repudiation of submission	–	–	–	–	–	–	*	–
Non-repudation of delivery	*	–	–	–	–	–	–	–
Message security labelling								
Message security labelling	*	*	*	*	*	*	*	*
Security management service								
Change credentials	–	*	–	*	*	*	*	–
Register	–	*	–	*	–	–	–	–
MS-register	–	*	–	–	–	–	–	–

* An asterisk under the heading of the form *X/Y* indicates that the service can be provided from a functional object of type *X* to one of type *Y*.

a) This service is provided by the recipient's MS to the originator's UA.

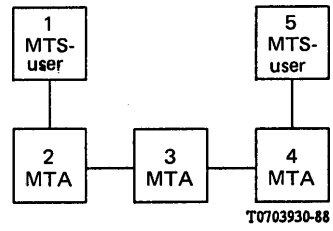


FIGURE 6/X.402

Simplified MHS functional model

10.2.1.2 *Proof of Submission security service*

This security service enables the originator of a message to obtain corroboration that it has been received by the MTS for delivery to the originally specified recipient(s).

This security service can be provided by using the Proof of Submission security element.

10.2.1.3 *Proof of Delivery security service*

This security service enables the originator of a message to obtain corroboration that it has been delivered by the MTS to its intended recipient(s).

This security service can be provided by using the Proof of Delivery security element.

10.2.2 *Secure Access Management security service*

The Secure Access Management security service is concerned with providing protection for resources against their unauthorised use. It can be divided into two components, namely the Peer Entity Authentication and the Security Context security services.

10.2.2.1 *Peer Entity Authentication security service*

This security service is provided for use at the establishment of a connection to confirm the identity of the connecting entity. It may be used on the links 1-2, 2-3, 3-4, or 4-5 in Figure 6/X.402 and provides confidence, at the time of usage only, that an entity is not attempting a masquerade or an unauthorised replay of a previous connection.

This security service is supported by the Authentication Exchange security element. Note that use of this security element may yield other data as a result of its operation that in certain circumstances can be used to support a Connection Confidentiality and/or a Connection Integrity security service.

10.2.2.2 *Security Context security service*

This security service is used to limit the scope of passage of messages between entities by reference to the Security Labels associated with messages. This security service is therefore closely related to the Message Security Labelling security service, which provides for the association of messages and Security Labels.

The Security Context security service is supported by the Security Context and the Register security elements.

10.2.3 *Data Confidentiality security services*

These security services provide for the protection of data against unauthorised disclosure.

10.2.3.1 *Connection Confidentiality security service*

The MHS does not provide a Connection Confidentiality security service. However, data for the invocation of such a security service in underlying layers may be provided as a result of using the Authentication Exchange security element to provide the Peer Entity Authentication security service. The security service may be required on any of links 1-2, 2-3, 3-4, or 4-5 in Figure 6/X.402.

10.2.3.2 *Content Confidentiality security service*

The Content Confidentiality security service provides assurance that the content of a message is only known to the sender and recipient of a message.

It may be provided using a combination of the Content Confidentiality and the Message Argument Confidentiality security elements. The Message Argument Confidentiality security element can be used to transfer a secret key which is used with the Content Confidentiality security element to encipher the message content. Using these security elements the service is provided from MTS-user 1 to MTS-user 5 in the figure, with the message content being unintelligible to MTAs.

10.2.3.3 *Message Flow Confidentiality security service*

This security service provides for the protection of information which might be derived from observation of message flow. Only a limited form of this security service is provided by the MHS.

The Double Enveloping Technique enables a complete message to become the content of another message. This could be used to hide addressing information from certain parts of the MTS. Used in conjunction with traffic padding (which is beyond the current scope of this Recommendation) this could be used to provide message flow confidentiality. Other elements of this service, such as routing control or pseudonyms, are also beyond the scope of this Recommendation.

10.2.4 *Data Integrity security services*

These security services are provided to counter active threats to the MHS.

10.2.4.1 *Connection Integrity security service*

The MHS does not provide a Connection Integrity security service. However, data for the invocation of such a security service in underlying layers may be provided by using the Authentication Exchange security element to provide the Peer Entity Authentication security service. The security service may be required on any of links 1-2, 2-3, 3-4, or 4-5 in Figure 6/X.402.

10.2.4.2 *Content Integrity security service*

This security service provides for the integrity of the contents of a single message. This takes the form of enabling the determination of whether the message content has been modified. This security service does not enable the detection of message replay, which is provided by the Message Sequence Integrity security service.

This security service can be provided in two different ways using two different combinations of security elements.

The Content Integrity security element together with the Message Argument Integrity security element and, in some cases, the Message Argument Confidentiality security element can be used to provide the security service to a message recipient, i.e., for communication from MTS-user 1 to MTS-user 5 in Figure 6/X.402. The Content Integrity security element is used to compute a Content Integrity Check as a function of the entire message content. Depending on the method used to compute the Content Integrity Check, a secret key may be required, which may be confidentially sent to the message recipient using the Message Argument Confidentiality security element. The Content Integrity Check is protected against change using the Message Argument Integrity security element. The integrity of any confidential message arguments is provided using the Message Argument Confidentiality security element.

The Message Origin Authentication security element can also be used to provide this security service.

10.2.4.3 *Message Sequence Integrity security service*

This security service protects the originator and recipient of a sequence of messages against re-ordering of the sequence. In doing so it protects against replay of messages.

This security service may be provided using a combination of the Message Sequence Integrity and the Message Argument Integrity security elements. The former provides a sequence number to each message, which may be protected against change by use of the latter. Simultaneous confidentiality and integrity of the Message Sequence Number may be provided by use of the Message Argument Confidentiality security element.

These security elements provide the service for communication from MTS-user 1 to MTS-user 5 in Figure 6/X.402, and not to the intermediate MTAs.

10.2.5 *Non-repudiation security services*

These security services provide irrevocable proof to a third party after the message has been submitted, sent, or delivered, that the submission, sending, or receipt did occur as claimed. Note that for this to function correctly, the security policy must explicitly cover the management of asymmetric keys for the purpose of non-repudiation services if asymmetric algorithms are being used.

10.2.5.1 *Non-repudiation of origin security service*

This security service provides the recipient(s) of a message with irrevocable proof of the origin of the message, its content, and its associated Message Security Label.

This security service can be provided in two different ways using two different combinations of security elements. Note that its provision is very similar to the provision of the (weaker) Content Integrity security service.

The Content Integrity security element together with the Message Argument Integrity security element and, in some cases, the Message Argument Confidentiality security element can be used to provide the service to a message recipient, i.e., for communication from MTS-user 1 to MTS-user 5 in Figure 6/X.402. The Content Integrity security element is used to compute a Content Integrity Check as a function of the entire message content. Depending on the method used to compute the Content Integrity Check, a secret key may be required, which may be confidentially sent to the message recipient using the Message Argument Confidentiality security element. The Content Integrity Check and, if required, the Message Security Label are protected against change and/or repudiation using the Message Argument Integrity security element. Any confidential message arguments are protected against change and/or repudiation using the Message Argument Confidentiality security element.

If the Content Confidentiality security service is not required, the Message Origin Authentication security element may also be used as a basis for this security service. In this case the security service may be provided to all elements of the MHS, i.e., for all of 1-5 in Figure 6/X.402.

10.2.5.2 *Non-repudiation of Submission security service*

This security service provides the originator of the message with irrevocable proof that the message was submitted to the MTS for delivery to the originally specified recipient(s).

This security service is provided using the Proof of Submission security element in much the same way as that security element is used to support the (weaker) Proof of Submission security service.

10.2.5.3 *Non-repudiation of Delivery security service*

This security service provides the originator of the message with irrevocable proof that the message was delivered to its originally specified recipient(s).

This security service is provided using the Proof of Delivery security element in much the same way as that security element is used to support the (weaker) Proof of Delivery security service.

10.2.6 *Message Security Labelling security service*

This security service allows Security Labels to be associated with all entities in the MHS, i.e., MTAs and MTS-users. In conjunction with the Security Context security service it enables the implementation of security policies defining which parts of the MHS may handle messages with specified associated Security Labels.

This security service is provided by the Message Security Label security element. The integrity and confidentiality of the label are provided by the Message Argument Integrity and the Message Argument Confidentiality security elements.

10.2.7 *Security management services*

A number of security management services are needed by the MHS. The only management services provided within Recommendation X.411 are concerned with changing credentials and registering MTS-user security labels.

10.2.7.1 *Change Credentials security service*

This security service enables one entity in the MHS to change the credentials concerning it held by another entity in the MHS. It may be provided using the Change Credentials security element.

10.2.7.2 *Register security service*

This security service enables the establishment at an MTA of the Security Labels which are permissible for one particular MTS-user. It may be provided using the Register security element.

10.2.7.3 *MS-register security service*

The security service enables the establishment of the security label which ar permissible for the MS-user.

10.3 *Security elements*

The following clauses describe the security elements available in the protocols described within Recommendation X.411 to support the security services in the MHS. These security elements relate directly to arguments in various services described in Recommendation X.411. The objective of this clause is to separate out each element of the Recommendation X.411 service definitions that relate to security, and to define the function of each of these identified security elements.

10.3.1 *Authentication security elements*

These security elements are defined in order to support authentication and integrity security services.

10.3.1.1 *Authentication exchange security element*

The Authentication Exchange security element is designed to authenticate, possibly mutually, the identity of an MTS-user to an MTA, an MTA to an MTA, an MTA to an MTA-user, an MS to a UA, or a UA to an MS. It is based on the exchange or use of secret data, either passwords, asymmetrically encrypted tokens, or symmetrically encrypted tokens. The result of the exchange is corroboration of the identity of the other party, and, optionally, the transfer of confidential data which may be used in providing the Connection Confidentiality and/or the Connection Integrity security service in underlying layers. Such an authentication is only valid for the instant that it is made and the continuing validity of the authenticated identity depends on whether the exchange of confidential data, or some other mechanism, is used to establish a secure communication path. The establishment and use of a secure communication path is outside the scope of this Recommendation.

This security element uses the Initiator Credentials argument and the Responder Credentials result of the MTS-bind, MS-bind and MTA-bind services. The transferred credentials are either passwords or tokens.

10.3.1.2 *Data Origin Authentication security elements*

These security elements are specifically designed to support data origin authentication services, although they may also be used to support certain data integrity services.

10.3.1.2.1 *Message Origin Authentication security element*

The Message Origin Authentication security element enables anyone who receives or transfers a message to authenticate the identity of the MTS-user that originated the message. This may mean the provision of the Message Origin Authentication or the Non-repudiation of Origin security service.

The security element involves transmitting, as part of the message, a Message Origin Authentication Check, computed as a function of the message content, the message Content Identifier, and the Message Security Label. If the Content Confidentiality security service is also required, the Message Origin Authentication Check is computed as a function of the enciphered rather than the unenciphered message content. By operating on the message content as conveyed in the overall message (i.e., after the optional Content Confidentiality security element), any MHS entity can check the overall message integrity without the need to see the plaintext message content. However, if the Content Confidentiality security service is used, the Message Origin Authentication security element cannot be used to provide the Non-repudiation of Origin security service.

The security element uses the Message Origin Authentication Check, which is one of the arguments of the Message Submission, Message Transfer, and Message Delivery services.

10.3.1.2.2 *Probe Origin Authentication security element*

Similar to the Message Origin Authentication security element, the Probe Origin Authentication security element enables any MTA to authenticate the identity of the MTS-user which originated a probe.

This security element uses the Probe Origin Authentication Check, which is one of the arguments of the Probe Submission service.

10.3.1.2.3 *Report Origin Authentication security element*

Similar to the Message Origin Authentication security element, the Report Origin Authentication security element enables any MTA or MTS-user who receives a report to authenticate the identity of the MTA which originated the report.

This security element uses the Report Origin Authentication Check, which is one of the arguments of the Report Delivery service.

10.3.1.3 *Proof of Submission security element*

This security element provides the originator of a message with the means to establish that a message was accepted by the MHS for transmission.

The security element is made up of two arguments: a request for Proof of Submission, sent with a message at submission time, and the Proof of Submission, returned to the MTS-user as part of the Message Submission results. The Proof of Submission is generated by the MTS, and is computed as a function of all the arguments of the submitted message, the Message Submission Identifier, and the Message Submission Time.

The Proof of Submission argument can be used to support the Proof of Submission security service. Depending on the security policy in force, it may also be able to support the (stronger) Non-repudiation of Submission security service.

The Proof of Submission Request is an argument of the Message Submission service. The Proof of Submission is one of the results of the Message Submission service.

10.3.1.4 *Proof of Delivery security element*

This security element provides the originator of a message with the means to establish that a message was delivered to the destination by the MHS.

The security element is made up of a number of arguments. The message originator includes a Proof of Delivery Request with the submitted message, and this request is delivered to each recipient with the message. A recipient may then compute the Proof of Delivery as a function of a number of arguments associated with the message. The proof of delivery is returned by the MTS to the message originator, as part of a report on the results of the original Message Submission.

The Proof of Delivery can be used to support the Proof of Delivery security service. Depending on the security policy in force, it may also be able to support the (stronger) Non-repudiation of Delivery security service.

The Proof of Delivery Request is an argument of the Message Submission, Message Transfer, and Message Delivery services. The Proof of Delivery is both one of the results of the Message Delivery service and one of the arguments of the Report Transfer and Report Delivery services.

Note – Non-receipt of a Proof of Delivery does not imply non-delivery.

10.3.2 *Secure Access Management security elements*

These security elements are defined in order to support the Secure Access Management security service and the security management services.

10.3.2.1 *Security Context security element*

When an MTS-user or an MTA binds to an MTA or MTS-user, the bind operation specifies the security context of the connection. This limits the scope of passage of messages by reference to the labels associated with messages. Secondly, the Security Context of the connection may be temporarily altered for submitted or delivered messages.

The Security Context itself consists of one or more Security Labels defining the sensitivity of interactions that may occur in line with the security policy in force.

Security Context is an argument of the MTS-bind and MTA-bind services.

10.3.2.2 *Register security element*

The Register security element allows the establishment at an MTA of an MTS-user's permissible security labels.

This security element is provided by the Register service. The Register service enables an MTS-user to change arguments, held by the MTS, relating to delivery of messages to that MTS-user.

10.3.2.3 *MS-Register security element*

The MS-Register security element allows the establishment of the MS-user's permissible security labels.

This security element is provided by the MS-Register service. The MS-Register services enables an MS-user to change arguments held by the MS relating to the retrieval of messages to that MS-user.

10.3.3 *Data Confidentiality security elements*

These security elements, based on the use of encipherment, are all concerned with the provision of confidentiality of data passed from one MHS entity to another.

10.3.3.1 *Content Confidentiality security element*

The Content Confidentiality security element provides assurance that the content of the message is protected from eavesdropping during transmission by use of an encipherment security element. The security element operates such that only the recipient and sender of the message know the plaintext message content.

The specification of the encipherment algorithm, the key used, and any other initialising data are conveyed using the Message Argument Confidentiality and the Message Argument Integrity security elements. The algorithm and key are then used to encipher or decipher the message contents.

The Content Confidentiality security element uses the Content Confidentiality Algorithm Identifier, which is an argument of the Message Submission, Message Transfer, and Message Delivery services.

10.3.3.2 *Message Argument Confidentiality security element*

The Message Argument Confidentiality security element provides for the confidentiality, integrity, and, if required, the irrevocability of recipient data associated with a message. Specifically, this data will comprise any cryptographic keys and related data that is necessary for the confidentiality and integrity security elements to function properly, if these optional security elements are invoked.

The security element operates by means of the Message Token. The data to be protected by the Message Argument Confidentiality security element constitutes the Encrypted Data within the Message Token. The Encrypted Data within the Message Token is unintelligible to all MTAs.

The Message Token is an argument of the Message Submission, Message Transfer, and Message Delivery services.

10.3.4 *Data Integrity security elements*

These security elements are provided to support the provision of data integrity, data authentication, and non-repudiation services.

10.3.4.1 *Content Integrity security element*

The Content Integrity security element provides protection for the content of a message against modification during transmission.

This security element operates by use of one or more cryptographic algorithms. The specification of the algorithm(s), the key(s) used, and any other initialising data are conveyed using the Message Argument Confidentiality and the Message Argument Integrity security elements. The result of the application of the algorithms and key is the Content Integrity Check, which is sent in the message envelope. The security element is only available to the recipient(s) of the message as it operates on the plaintext message contents.

If the Content Integrity Check is protected using the Message Argument Integrity security element then, depending on the prevailing security policy, it may be used to help provide the Non-repudiation of Origin security service.

The Content Integrity Check is an argument of the Message Submission, Message Transfer, and Message Delivery services.

10.3.4.2 *Message Argument Integrity security element*

The Message Argument Integrity security element provides for the integrity, and, if required, the irrevocability of certain arguments associated with a message. Specifically, these arguments may comprise any selection of the Content Confidentiality Algorithm Identifier, the Content Integrity Check, the Message Security Label, the Proof of Delivery Request, and the Message Sequence Number.

The security element operates by means of the Message Token. The data to be protected by the Message Argument Integrity security element constitutes the signed-data within the Message Token.

The Message Token is an argument of the Message Submission, Message Transfer, and Message Delivery services.

10.3.4.3 *Message Sequence Integrity security element*

The Message Sequence Integrity security element provides protection for the sender and recipient of a message against receipt of messages in the wrong order, or duplicated messages.

A Message Sequence Number is associated with an individual message. This number identifies the position of a message in a sequence from one originator to one recipient. Therefore each originator-recipient pair requiring to use this security element will have to maintain a distinct sequence of message numbers. This security element does not provide for initialisation or synchronisation of Message Sequence Numbers.

10.3.5 *Non-repudiation security elements*

There are no specific Non-repudiation security elements defined in Recommendation X.411. The non-repudiation services may be provided using a combination of other security elements.

10.3.6 *Security Label security elements*

These security elements exist to support security labelling in the MHS.

10.3.6.1 *Message Security Label security element*

Messages may be labelled with data as specified in the prevailing security policy. The Message Security Label is available for use by intermediate MTAs as part of the overall security policy of the system.

A Message Security Label may be sent as a message argument, and may be protected by the Message Argument Integrity or the Message Origin Authentication security element, in the same manner as other message arguments.

Alternatively, if both confidentiality and integrity are required, the Message Security Label may be protected using the Message Argument Confidentiality security element. In this case the Message Security Label so protected is an originator-recipient argument, and may differ from the Message Security Label in the message envelope.

10.3.7 *Security Management security elements*

10.3.7.1 *Change Credentials security element*

The Change Credentials security element allows the credentials of an MTS-user or an MTA to be updated.

The security element is provided by the MTS Change Credentials service.

10.3.8 *Double Enveloping Technique*

Additional protection may be provided to a complete message, including the envelope parameters, by the ability to specify that the content of a message is itself a complete message, i.e., a Double Enveloping Technique is available.

This technique is available though the use of the Content Type argument which makes it possible to specify that the content of a message is an Inner Envelope. This Content Type means that the content is itself a message (envelope and content) for forwarding by the recipient named on the outer envelope to the recipient named on the Inner Envelope.

The Content Type is an argument of the Message Submission, Message Transfer, and Message Delivery services.

SECTION 3 – CONFIGURATIONS

11 Overview

This section specifies how one can configure the MHS to satisfy any of a variety of functional, physical, and organizational requirements.

This section covers the following topics:

a) functional configurations;

b) physical configurations;

c) organizational configurations;

d) the *Global MHS.*

12 Functional configurations

This clause specifies the possible functional configurations of the MHS. The variety of such configurations results from the presence or absence of the Directory, and from whether a direct user employs an MS.

12.1 *Regarding the Directory*

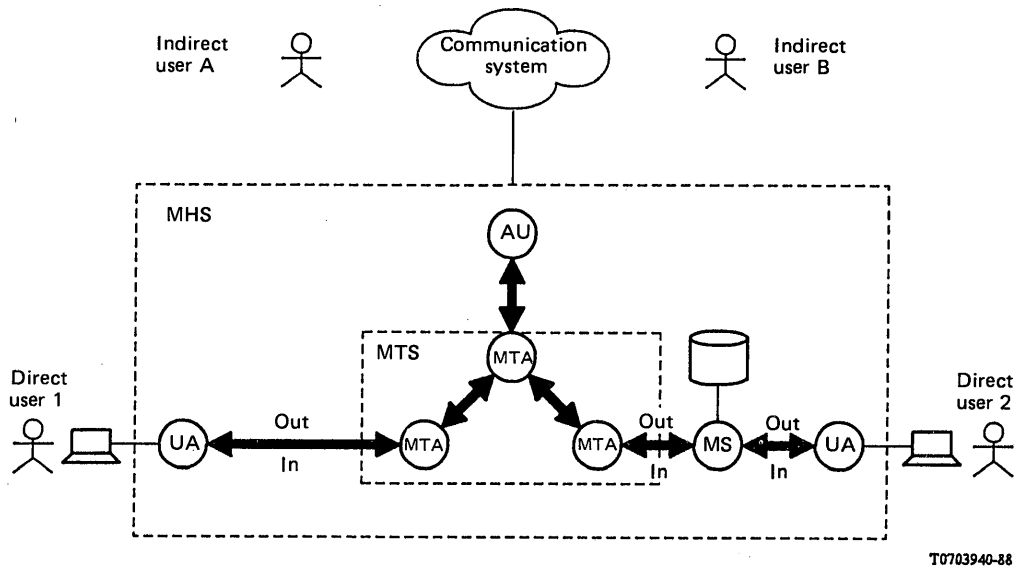
With respect to the Directory, the MHS can be configured for a particular user, or a collection of users (e.g., see § 14.1), in either of two ways: with or without the Directory. A user without access to the Directory may lack the capabilities described in Section 5.

Note – A partially, rather than fully, interconnected directory may exist for an interim period during which the (global) Directory made possible by Recommendations for Directories is under construction.

12.2 *Regarding the message store*

With respect to the MS, the MHS can be configured for a particular direct user in either of two ways: with or without an MS. A user without access to an MS lacks the capabilities of message storage. A user in such circumstances depends upon his UA for the storage of information objects, a capability that is a local matter.

The two functional configurations identified above are depicted in Figure 7/X.402 which also illustrates one possible configuration of the MTS, and its linkage to another communication system via an AU. In the figure, user 2 is equipped with an MS while user 1 is not.



Note – Whilde the users depicted in the Figure are people, the Figure applies with equal force and validity to users of other kinds.

FIGURE 7/X.402

Functional configurations regarding the MS

13 Physical configurations

This clause specifies the possible physical configurations of the MHS, i.e., how the MHS can be realized as a set of interconnected computer systems. Because the number of configurations is unbounded, the clause describes the kinds of messaging systems from which the MHS is assembled, and identifies a few important representative configurations.

13.1 *Messaging systems*

The building blocks used in the physical construction of the MHS are called *messaging systems.* A **messaging system** is a computer system (possibly but not necessarily an open system) that contains, or realizes, one or more functional objects.

Messaging systems are of the types depicted in Figure 8/X.402.

The types of messaging system, depicted in Figure 8/X.402, are listed in the first column of Table 8/X.402. For each type listed, the second column indicates the kinds of functional object – UAs, MSs, MTAs, and AUs – that may be present in such a messaging system, whether their presence is mandatory or optional, and whether just one or possibly several of them may be present in the messaging system. The table is divided into two sections. Messaging systems of the types in the first section are dedicated to single users, those of the types in the second can (but need not) serve multiple users.

Table 8/X.402 is divided into two sections. Messaging systems of the types in the first section are dedicated to single users, those of the types in the second can (but need not) serve multiple users.

Note – The following major principles governed the admission of messaging system types:

a) An AU and the MTA with which it interacts are typically co-located because no protocol to govern their interaction is standardized.

b) An MTA is typically co-located with multiple UAs or MSs because, of the standardized protocols, only that for transfer simultaneously conveys a message to multiple recipients. The serial delivery of a message to multiple recipients served by a messaging system, which the delivery protocol would require, would be inefficient.

c) No purpose is served by co-locating several MTAs in a messaging system because a single MTA serves multiple users, and the purpose of an MTA is to convey objects between, not within such systems. (This is not intended to exclude the possibility of several MTA-related processes co-existing within a single computer system.)

d) The co-location of an AU with an MTA does not affect that system's behaviour with respect to the rest of the MHS. A single messaging system type, therefore, encompasses the AU's presence and absence.

The messaging system types, summarized in Table 8/X.402, are individually defined and described below.

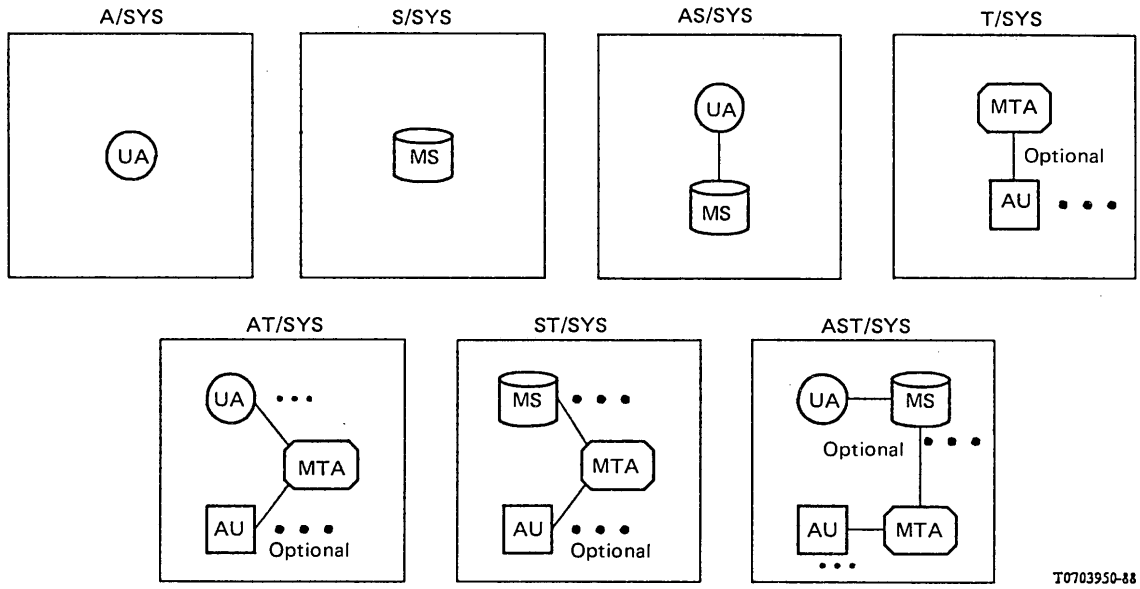


FIGURE 8/X.402

Messaging system types

TABLE 8/X.402

Messaging systems

Messaging system	Functional objects			
	UA	MS	MTA	AU
A/SYS	1	–	–	–
S/SYS	–	1	–	–
AS/SYS	1	1	–	–
T/SYS	–	–	1	[M]
AT/SYS	M	–	1	[M]
ST/SYS	–	M	1	[M]
AST/SYS	M	M	1	[M]

M Multiple

[. . .] Optional

13.1.1 *Access systems*

An **access system (A/SYS)** contains one UA and neither an MS, an MTA, nor an AU.

An A/SYS is dedicated to a single user.

13.1.2 *Storage systems*

A **storage system (S/SYS)** contains one MS and neither a UA, an MTA, nor an AU.

An S/SYS is dedicated to a single user.

13.1.3 *Access and storage systems*

An **access and storage system (AS/SYS)** contains one UA, one MS, and neither an MTA nor an AU.

An AS/SYS is dedicated to a single user.

13.1.4 *Transfer systems*

A **transfer system (T/SYS)** contains one MTA; optionally, one or more AUs; and neither a UA nor an MS.

A T/SYS can serve multiple users.

13.1.5 *Access and transfer systems*

An **access and transfer system (AT/SYS)** contains one or more UAs; one MTA; optionally, one or more AUs; and no MS.

An AT/SYS can serve multiple users.

13.1.6 *Storage and transfer systems*

A **storage and transfer system (ST/SYS)** contains one or more MSs; one MTA; optionally, one or more AUs; and no UA.

An ST/SYS can serve multiple users.

13.1.7 *Access, storage, and transfer systems*

An **access, storage, and transfer system (AST/SYS)** contains one or more UAs; one or more MSs; one MTA; and optionally, one or more AUs.

An AST/SYS can serve multiple users.

13.2 *Representative configurations*

Messaging systems can be combined in various ways to form the MHS. The possible physical configurations are unbounded in number and thus cannot be enumerated. Several important representative configurations, however, are described below and in Figure 9/X.402.

13.2.1 *Fully centralized*

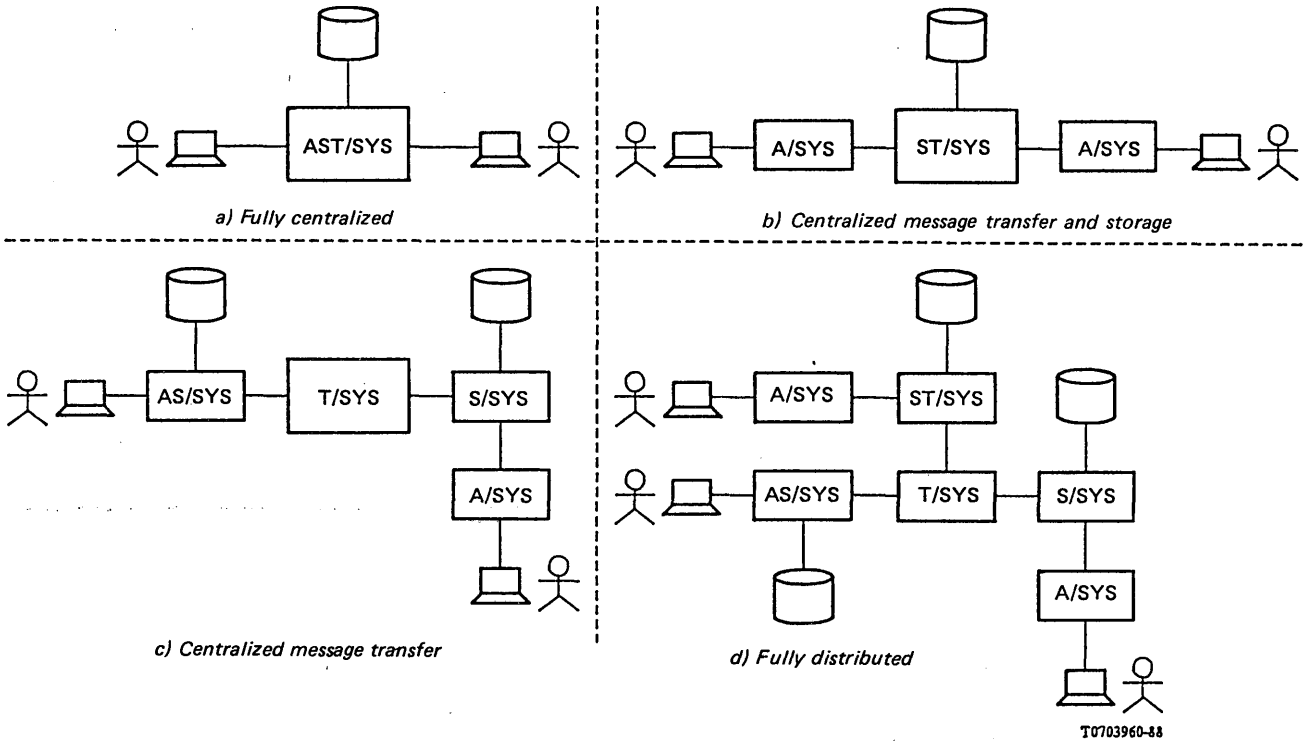
The MHS may be fully centralized [panel a) of Figure 9/X.402]. This design is realized by a single AST/SYS which contains functional objects of all kinds and which can serve multiple users.

13.2.2 *Centralized message transfer and storage*

The MHS may provide both message transfer and message storage centrally but user access distributedly [panel b) of Figure 9/X.402]. This design is realized by a single ST/SYS and, for each user, an A/SYS.

13.2.3 *Centralized message transfer*

The MHS may provide message transfer centrally but message storage and user access distributedly [panel c) of Figure 9/X.402]. This design is realized by a single T/SYS and, for each user, either an AS/SYS alone or an S/SYS and an associated A/SYS.



Note 1 – While the users depicted in the Figure are people, the Figure applies with equal force and validity to users of other kinds.
Note 2 – Besides the physical configurations that result from the "pure" approaches below, many "hybrid" configurations can be constructed.

FIGURE 9/X.402

Representative physical configurations

13.2.4 *Fully distributed*

The MHS may provide even message transfer distributedly [panel d) of Figure 9/X.402]. This design involves multiple ST-SYSs or T-SYSs.

14 Organizational configurations

This clause specifies the possible organizational configurations of the MHS, i.e., how the MHS can be realized as interconnected but independently managed sets of messaging systems (which are themselves interconnected). Because the number of configurations is unbounded, the clause describes the kinds of *management domains* from which the MHS is assembled, and identifies a few important representative configurations.

14.1 *Management domains*

The primary building blocks used in the organizational construction of the MHS are called *management domains.* A **management domain** (MD) (or **domain**) is a set of messaging systems – at least one of which contains, or realizes, an MTA – that is managed by a single organization.

The above does not preclude an organization from managing a set of messaging systems (e.g., a single A/SYS) that does not qualify as an MD for lack of an MTA. Such a collection of messaging systems, a secondary building block used in the MHS' construction, "attaches" to an MD.

MDs are of several types which are individually defined and described below.

14.1.1 *Administration management domains*

An **administration management domain (ADMD)** comprises messaging systems managed by an Administration. The major technical distinction between an ADMD and a *PRMD* is that the former is positioned above the latter in the MHS' hierarchical addressing (see § 18) and routing (see § 19) regimes.

Note – An ADMD provides Message Handling to the public.

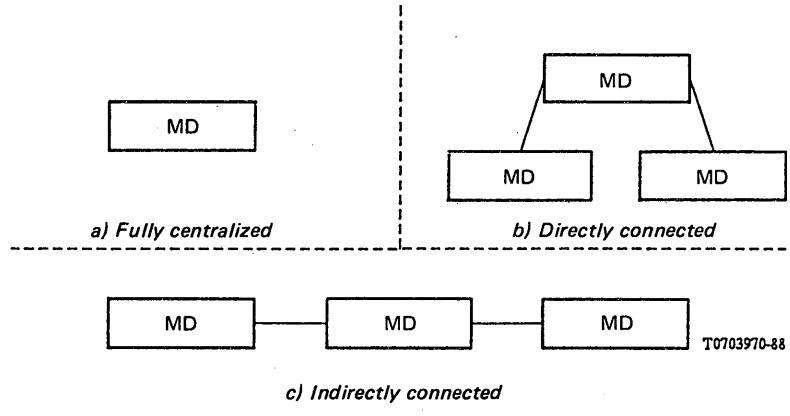
14.1.2 *Private management domains*

A **private management domain (PRMD)** comprises messaging systems managed by an organization other than an Administration. The major technical distinction between a PRMD and an ADMD is that the former is positioned below the latter in the MHS' hierarchical addressing (see § 18) and routing (see § 19) regimes.

Note – A PRMD provides message handling, e.g., to the employees of a company, or to those employees at a particular company site.

14.2 *Representative configurations*

MDs can be combined in various ways to form the MHS. The possible organizational configurations are unbounded in number and thus cannot be enumerated. Several important representative configurations, however, are described below and in Figure 10/X.402.



Note – Besides the organizational configurations that result from the "pure" approaches below, many "hybrid" configurations can be constructed.

FIGURE 10/X.402

Representative organizational configurations

14.2.1 *Fully centralized*

The entire MHS may be managed by one organization [panel a) of Figure 10/X.402]. This design is realized by a single MD.

14.2.2 *Directly connected*

The MHS may be managed by several organizations, the messaging systems of each connected to the messaging systems of all of the others [panel b) of Figure 10/X.402]. This design is realized by multiple MDs interconnected pair-wise.

14.2.3 *Indirectly connected*

The MHS may be managed by several organizations, the messaging systems of one serving as intermediary between the messaging systems of the others [panel c) of Figure 10/X.402]. This design is realized by multiple MDs one of which is interconnected to all of the others.

15 The Global MHS

A major purpose of this Recommendation and others in the set is to enable the construction of the Global MHS, an MHS providing both intra- and inter-organizational, and both intra- and international message handling world-wide.

The Global MHS almost certainly encompasses the full variety of functional configurations specified in § 12.

The physical configuration of the Global MHS is a hybrid of the pure configurations specified in § 13, extremely complex and highly distributed physically.

The organizational configuration of the Global MHS is a hybrid of the pure configurations specified in § 14, extremely complex and highly distributed organizationally.

Figure 11/X.402 gives an example of possible interconnections. It does not attempt to identify all possible configurations. As depicted, ADMDs play a central role in the Global MHS. By interconnecting to one another internationally, they provide an international message transfer backbone. Depending upon national regulations, by interconnecting to one another domestically, they may also provide domestic backbones joined to the international backbone. ADMDs also serve as primary naming authorities in the assignment of *O/R addresses* to users and DLs.

PRMDs play a peripheral role in the Global MHS, being connected to the ADMD backbone which serves as an intermediary between them.

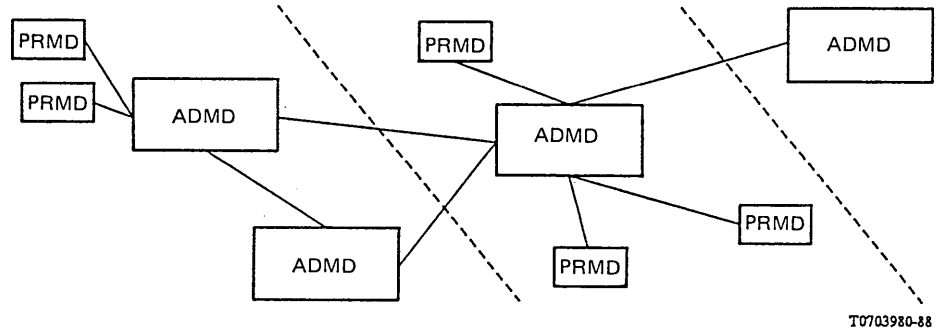


FIGURE 11/X.402

The global MHS

SECTION 4 – NAMING, ADDRESSING, AND ROUTING

16 Overview

This section describes the naming and addressing of users and DLs and the routing of information objects to them.

This section covers the following topics:

a) Naming;

b) Addressing;

c) Routing

17 Naming

This paragraph specifies how users and DLs are named for the purposes of message handling in general and message transfer in particular. It defines *O/R names* and describes the role that Directory names play in them.

When it directly submits a message or probe, a UA or MS identifies its potential recipients to the MTS. When the MTS delivers a message, it identifies the originator to each recipient's UA or MS. *O/R names* are the data structures by means of which such identification is achieved.

17.1 *Directory names*

A Directory name is one component of an *O/R name.* A Directory name identifies an object to the Directory. By presenting such a name to the Directory, the MHS can access a user's or DL's Directory entry. From that entry the MTS can obtain, e.g., the user's or DL's *O/R address.*

Not every user or DL is registered in the Directory and, therefore, not every user or DL possesses a Directory name.

Note 1 – Many users and DLs will lack Directory names until the Directory is widely available as an adjunct to the MHS. Many indirect users (e.g., postal patrons) will lack such names until the Directory is widely available as an adjunct to other communication systems.

Note 2 – Users and DLs may be assigned Directory names even before a fully interconnected, distributed Directory has been put in place by pre-establishing the naming authorities upon which the Directory will eventually depend.

Note 3 – The typical Directory name is more user-friendly and more stable than the typical *O/R address* because the latter is necessarily couched in terms of the organizational or physical structure of the MHS while the former need not be. Therefore, it is intended that over time, Directory names become the primary means by which users and DLs are identified outside the MTS (i.e. by other users), and that the use of *O/R address* be largely confined to the MTS (i.e., to use by MTAs).

17.2 *O/R names*

Every user or DL has one or more *O/R names.* An **O/R name** is an identifier by means of which a user can be designated as the originator, or a user or DL designated as a potential recipient of a message or probe. An O/R name distinguishes one user or DL from another and may also identify its point of access to the MHS.

An O/R name comprises a Directory name, an *O/R address,* or both. If present, the Directory name (if valid) unambiguously identifies the user or DL (but is not necessarily the only name that would do so). If present, the *O/R address* does the same and more (again see § 18.5).

At direct submission, the UA or MS of the originator of a message or probe may include either or both components in each O/R name it supplies. If the *O/R address* is omitted, the MTS obtains it from the Directory using the Directory name. If the Directory name is omitted, the MTS does without it. If both are included, the MTS relies firstly upon the *O/R address.* Should it determine that the *O/R address* is invalid (e.g., obsolete), it proceeds as if the *O/R address* had been omitted, relying upon the Directory name.

At delivery the MTS includes an *O/R address* and possibly a Directory name in each O/R name it supplies to a message's recipient or to the originator of a report's subject message or probe. The Directory name is included if the originator supplied it or if it was specified as the the member of an expanded DL.

Note – Redirection or DL expansion may cause the MTS to convey to a UA or MS at delivery, O/R names the UA or MS did not supply at direct submission.

18 Addressing

This paragraph specifies how users and DLs are addressed. It defines *O/R addresses,* describes the structure of the *attribute lists* from which they are constructed, discusses the character sets from which individual *attributes* are composed, gives rules for determining that two *attribute lists* are equivalent and for the inclusion of conditional *attributes* in such lists, and defines the *standard attributes* that may appear in them.

To convey a message, probe, or report to a user, or to expand a DL specified as a potential recipient of a message or probe, the MTS must locate the user or DL relative to its own physical and organizational structures. *O/R addresses* are the data structures by means of which all such location is accomplished.

18.1 *Attribute lists*

The *O/R addresses* of both users and DLs are attribute lists. An **attribute list** is an ordered set of *attributes.*

An **attribute** is an information item that describes a user or DL and that may also locate it in relation to the physical or organizational structure of the MHS (or the network underlying it).

An attribute has the following parts:

a) **attribute type (or type)**: An identifier that denotes a class of information (e.g., personal names).

b) **attribute value (or value)**: An instance of the class of information the attribute type denotes (e.g., a particular personal name).

Attributes are of the following two kinds:

a) **standard attribute**: An attribute whose type is bound to a class of information by this Recommendation.

The value of every standard attribute except terminal-type is either a string or a collection of strings.

b) **domain-defined attribute**: An attribute whose type is bound to a class of information by an MD.

Both the type and value of every domain-defined attribute are strings or collections of strings.

Note – The widespread use of standard attributes produces more uniform and thus more user-friendly O/R addresses. However, it is anticipated that not all MDs will be able to employ such attributes immediately. The purpose of domain-defined attributes is to permit an MD to retain its existing, native addressing conventions for a time. It is intended, however, that all MDs migrate toward the use of standard attributes, and that domain-defined attributes be used only for an interim period.

18.2 *Character sets*

Standard attribute values and domain-defined attribute types and values are constructed from numeric, printable, and teletex strings as follows:

a) The type or value of a particular domain-defined attribute may be a printable string, a teletex string, or both. The same choice shall be made for both the type and value.

b) The kinds of strings from which standard attribute values may be constructed and the manner of construction (e.g., as one string or several) vary from one attribute to another (see § 18.3).

The value of an attribute comprises strings of one of the following sets of varieties depending upon its type: numeric only, printable only, numeric and printable, and printable and teletex. With respect to this, the following rules govern each instance of communication:

a) Wherever both numeric and printable strings are permitted, strings of either variety (but not both) may be supplied equivalently.

b) Wherever both printable and teletex strings are permitted, strings of either or both varieties may be supplied, but printable strings shall be supplied as a minimum whenever attributes are conveyed internationally. If both printable and teletex strings are supplied, the two should convey the same information so that eiher of them can be safely ignored upon receipt.

The length of each string and of each sequence of strings in an attribute shall be limited as indicated in the more detailed (i.e., ASN.1) specification of attributes in Recommendation X.411.

Note 1 – Teletex strings are permitted in attribute values to allow inclusion, e.g., of the accented characters commonly used in many countries.

Note 2 – Not all input/output devices permit the entry and display, e.g., of accented characters. printable strings are required internationally to ensure that such device limitations do not prevent communication.

18.3 *Standard attributes*

The standard attribute types are listed in the first column of Table 9/X.402. For each listed type, the second column indicates the character sets – numeric, printable, and teletex – from which attribute values may be drawn.

Table 9/X.402 has three sections. Attribute types in the first are of a general nature, those in the second have to do with *routing to* a PDS, and those in the third have to do with *addressing within* a PDS.

TABLE 9/X.402

Standard attributes

Standard attribute type	Character sets		
	NUM	PRT	TTX
General			
administration-domain-name	×	×	–
common-name	–	×	×
country-name	×	×	–
network-address	× [a]	–	–
numeric-user-identifier	×	–	–
organization-name	–	×	×
organizational-unit-names	–	×	×
personal-name	–	×	×
private-domain-name	×	×	–
terminal-identifier	–	×	–
terminal-type	–	–	–
Postal routing			
physical-delivery-service-name	–	×	–
physical-delivery-country-name	×	×	–
postal-code	×	×	–
Postal addressing			
extension-postal-O/R-address-components	–	×	×
extension-physical-delivery-address-components	–	×	×
local-postal-attributes	–	×	×
physical-delivery-office-name	–	×	×
physical-delivery-office-number	–	×	×
physical-delivery-organization-name	–	×	×
physical-delivery-personal-name	–	×	×
post-office-box-address	–	×	×
poste-restante-address	–	×	×
street-address	–	×	×
unformatted-postal-address	–	×	×
unique-postal-name	–	×	×

NUM Numeric

PRT Printable

TTX Teletex

× Permitted

[a] Under prescribed circumstances a sequence of octet strings

The standard attribute types, summarized in Table 9/X.402, are individually defined and described below.

18.3.1 *Administration-domain-name*

An **administration-domain-name** is a standard attribute that identifies an ADMD relative to the country denoted by a country-name.

The value of an administration-domain-name is a numeric or printable string chosen from a set of such strings that is administered for this purpose by the country alluded to above.

Note – The attribute value comprising a single space (" ") shall be reserved for the following purpose. If permitted by the country denoted by the country-name attribute, a single space shall designate any (i.e., all) ADMDs within the country. This affects both the identification of users within the country and the routing of messages, probes, and reports to and among the ADMDs of that country. Regarding the former, it requires that the O/R addresses of users within the country be chosen so as to ensure their unambiguousness, even in the absence of the actual names of the users' ADMDs. Regarding the latter, it permits both PRMDs within, and ADMDs outside of the country, to route messages, probes, and reports to any of the ADMDs within the country indiscriminantly, and requires that the ADMDs within the country interconnect themselves in such a way that the messages, probes, and reports are conveyed to their destinations.

18.3.2 *Common-name*

A **common-name** is a standard attribute that identifies a user or DL relative to the entity denoted by another attribute (e.g., an organization-name).

The value of a common-name is a printable string, teletex string, or both. Whether printable or teletex, the string is chosen from a set of such strings that is administered for this purpose (and perhaps others) by the entity alluded to above.

Note – Among many other possibilities, a common-name might identify an organizational role (e.g., "Director of Marketing").

18.3.3 *Country-name*

A **country-name** is a standard attribute that identifies a country.

The value of a country-name is a numeric string that gives one of the numbers assigned to the country by Recommendation X.121, or a printable string that gives the character pair assigned to the country by ISO 3166.

18.3.4 *Extension-postal-O/R-address-components*

An **extension-postal-O/R-address-components** is a standard attribute that provides, in a postal address, additional information necessary to identify the addressee (e.g., an organizational unit).

The value of an extension-O/R-address-components is a printable string, teletex string, or both.

18.3.5 *Extension-physical-delivery-address-components*

An **extension-physical-delivery-address-components** is a standard attribute that specifies, in a postal address, additional information necessary to identify the exact point of delivery (e.g., room and floor numbers in a large building).

The value of an extension-physical-delivery-address-components is a printable string, teletex string, or both.

18.3.6 *Local-postal-attributes*

A **local-postal-attributes** is a standard attribute that identifies the locus of distribution, other than that denoted by a physical-delivery-office-name attribute (e.g., a geographical area), of a user's physical messages.

The value of a local-postal-attributes is a printable string, teletex string, or both.

18.3.7 *Network-address*

A **network-address** is a standard attribute that gives the network address of a terminal.

The value of a network-address is any one of the following:

a) a numeric string governed by Recommendation X.121;

b) two numeric strings governed by Recommendations E.163 and E.164;

c) a PSAP address.

Note – Among the strings admitted by Recommendation X.121 is a telex number preceded by the telex escape digit (8).

18.3.8 *Numeric-user-identifier*

A **numeric-user-identifier** is a standard attribute that numerically identifies a user relative to the ADMD denoted by an administration-domain-name.

The value of a numeric-user-identifier is a numeric string chosen from a set of such strings that is administered for this purpose by the ADMD alluded to above.

18.3.9 *Organization-name*

An **organization-name** is a standard attribute that identifies an organization. As a national matter, this identification may be either relative to the country denoted by a country-name (so that organization names are unique within the country), or relative to the MD identified by a private-domain-name, or an administration-domain-name, or both.

The value of an organization-name is a printable string, teletex string, or both. Whether printable or teletex, the string is chosen from a set of such strings that is administered for this purpose (and perhaps others) by the country or MD alluded to above.

Note – In countries choosing country-wide unique organization-names, a national registration authority for organization-names is required.

18.3.10 *Organizational-unit-names*

An **organizational-unit-names** is a standard attribute that identifies one or more units (e.g., divisions or departments) of the organization denoted by an organization-name, each unit but the first being a sub-unit of the units whose names precede it in the attribute.

The value of an organizational-unit-names is an ordered sequence of printable strings, an ordered sequence of teletex strings, or both. Whether printable or teletex, each string is chosen from a set of such strings that is administered for this purpose (and perhaps others) by the organization (or encompassing unit) alluded to above.

18.3.11 *Physicel-delivery-service-name*

A **physical-delivery-service-name** is a standard attribute that identifies a PDS relative to the ADMD denoted by an administration-domain-name.

The value of a physical-delivery-service-name is a printable string chosen from a set of such strings that is administered for this purpose by the ADMD alluded to above.

18.3.12 *Personal-name*

A **personal-name** is a standard attribute that identifies a person relative to the entity denoted by another attribute (e.g., an organization-name).

The value of a personal-name comprises the following four pieces of information, the first mandatory, the others optional:

a) the person's surname;

b) the person's given name;

c) the initials of all of his names but his surname;

d) his generation (e.g., "Jr").

The above information is supplied as printable strings, teletex strings, or both.

18.3.13 *Physical-delivery-country-name*

A **physical-delivery-country-name** is a standard attribute that identifies the country in which a user takes delivery of physical messages.

The value of a physical-delivery-country-name is subject to the same constraints as is the value of a country-name.

18.3.14 *Physical-delivery-office-name*

A **physical-delivery-office-name** is a standard attribute that identifies the city, village, etc. in which is situated the post office through which a user takes delivery of physical messages.

The value of a physical-delivery-office-name is a printable string, teletex string, or both.

18.3.15 *Physical-delivery-office-number*

A **physical-delivery-office-number** is a standard attribute that distinguishes among several post offices denoted by a single physical-delivery-office-name.

The value of a physical-delivery-office-number is a printable string, teletex string, or both.

18.3.16 *Physical-delivery-organization-name*

A **physical-delivery-organization-name** is a standard attribute that identifies a postal patron's organization.

The value of a physical-delivery-organization-name is a printable string, teletex string, or both.

18.3.17 *Physical-delivery-personal-name*

A **physical-delivery-personal-name** is a standard attribute that identifies a postal patron.

The value of a physical-delivery-personal-name is a printable string, teletex string, or both.

18.3.18 *Post-office-box-address*

A **post-office-box-address** is a standard attribute that specifies the number of the post office box by means of which a user takes delivery of physical messages.

The value of a postal-code is a numeric or printable string chosen from the set of such strings that is maintained and standardized for this purpose by the postal administration of the country identified by a physical-delivery-country-name attribute.

18.3.19 *Postal-code*

A **postal-code** is a standard attribute that specifies the postal code for the geographical area in which a user takes delivery of physical messages.

The value of a postal-code is a numeric or printable string chosen from the set of such strings that is maintained and standardized for this purpose by the postal administration of the country identified by a physical-delivery-country-name attribute.

18.3.20 *Poste-restante-address*

A **poste-restante-address** is a standard attribute that specifies the code that a user gives to a post office in order to collect the physical messages that await delivery to him.

The value of a poste-restante-address is a printable string, teletex string, or both chosen from the set of such strings assigned for this purpose by the post office denoted by a physical-delivery-office-name attribute.

18.3.21 *Private-domain-name*

A **private-domain-name** is a standard attribute that identifies a PRMD. As a national matter, this identification may be either relative to the country denoted by a country-name (so that PRMD names are unique within the country), or relative to the ADMD identified by an administration-domain-name.

The value of a private-domain-name is a numeric or printable string chosen from a set of such strings that is administered for this purpose by the country or ADMD alluded to above.

Note – In countries choosing country-wide unique PRMD names, a national registration authority for private-domain-names is required.

18.3.22 *Street-address*

A **street-address** is a standard attribute that specifies the street address (e.g., house number and street name and type (e.g., "Road")) at which a user takes delivery of physical messages.

The value of a street-address is a printable string, teletex string, or both.

18.3.23 *Terminal-identifier*

A **terminal-identifier** is a standard attribute that gives the terminal identifier of a terminal (e.g., a Telex answer back or a Teletex terminal identifier).

The value of a terminal-identifier is a printable string.

18.3.24 *Terminal-type*

A **terminal-type** is a standard attribute that gives the type of a terminal.

The value of a terminal-type is any one of the following: *telex, teletex, G3 facsimile, G4 facsimile, IA5 terminal,* and *videotex.*

18.3.25 *Unformatted-postal-address*

An **unformatted-postal-address** is a standard attribute that specifies a user's postal address in free form.

The value of an unformatted-postal address is a sequence of printable strings, each representing a line of text, a single teletex string, lines being separated as prescribed for such strings; or both.

18.3.26 *Unique-postal-name*

A **unique-postal-name** is a standard attribute that identifies the point of delivery, other than that denoted by a street-address, post-office-box-address, or poste-restante-address, (e.g., a building or hamlet) of a user's physical messages.

The value of a unique-postal-name is a printable string, teletex string, or both.

18.4 *Attribute list equivalence*

Several O/R addresses, and thus several attribute lists, may denote the same user or DL. This multiplicity of O/R addresses results in part (but not in full) from the following attribute list equivalence rules:

a) The relative order of standard attributes is insignificant.

b) Where the value of a standard attribute may be a numeric string or an equivalent printable string, the choice between them shall be considered insignificant.

Note – This rule applies even to the country-name standard attribute, where the choice between X.121 or ISO 3166 forms shall be considered insignificant, where X.121 allocates more than one number to a country, the significance of which number is used has not been standardized by this Recommendation.

c) Where the value of a standard attribute may be a printable string, an equivalent teletex string, or both, the choice between the three possibilities shall be considered insignificant.

d) Where the value of a standard attribute may contain letters, the cases of those letter shall be considered insignificant.

e) In a domain-defined attribute type or value, or in a standard attribute value, all leading, all trailing, and all but one consecutive embedded spaces shall be considered insignificant.

Note 1 – An MD may impose additional equivalence rules upon the attributes it assigns to its own users and DLs. It might define, e.g., rules concerning punctuation characters in attribute values, the case of letters in such values, or the relative order of domain-defined attributes.

Note 2 – As a national matter, MDs may impose additional equivalence rules regarding standard attributes whose values are given as teletex strings, in particular, the rules for deriving the equivalent printable strings.

18.5 *O/R address forms*

Every user or DL is assigned one or more O/R addresses. An **O/R address** is an attribute list that distinguishes one user from another and identifies the user's point of access to the MHS or the DL's expansion point.

An O/R address may take any of the forms summarized in Table 10/X.402. The first column of the table identifies the attributes available for the construction of O/R addresses. For each O/R address form, the second column indicates the attributes that may appear in such O/R addresses and their grades (see also § 18.6).

Table 10/X.402 has four sections. Attribute types in the first are those of a general nature, attribute types in the second and third those specific to physical delivery. The fourth section encompasses domain-defined attributes.

The forms of O/R address, summarized in Table 10/X.402 are individually defined and described below.

18.5.1 *Mnemonic O/R address*

A **mnemonic O/R address** is one that mnemonically identifies a user or DL. It identifies an ADMD, and a user or DL relative to it.

A mnemonic O/R address comprises the following attributes:

a) one country-name and one administration-domain-name, which together identify an ADMD;

b) one private-domain-name, one organization-name, one organizational-unit-names, one personal-name or common-name, or a combination of the above; and optionally one or more domain-defined attributes; which together identify a user or DL relative to the ADMD in item a) above.

18.5.2 *Numeric O/R address*

A **numeric O/R address** is one that numerically identifies a user. It identifies an ADMD, and a user relative to it.

A numeric O/R address comprises the following attributes:

a) one country-name and one administration-domain-name, which together identify an ADMD;

b) one numeric-user-identifier and, conditionally, one private-domain-name, which together identify the user relative to the ADMD in item a above;

c) conditionally, one or more domain-defined attributes which provide information additional to that which identifies the user.

18.5.3 *Postal O/R address*

A **postal O/R address** is one that identifies a user by means of its postal address. It identifies the PDS through which the user is to be accessed and gives the user's postal address.

The following kinds of postal O/R address are distinguished:

a) **formatted**;: Said of a postal O/R address that specifies a user's postal address by means of several attributes. For this form of postal O/R address, this Recommendation prescribes the structure of postal addresses in some detail;

b) **unformatted**;: Said of a postal O/R address that specifies a user's postal address in a single attribute. For this form of postal O/R address, this Recommendation largely does not prescribe the structure of postal addresses.

TABLE 10/X.402

Forms of O/R address

Attribute type	O/R address forms				
	MNEM	NUMR	POST		TERM
			F	U	
General					
administration-domain-name	M	M	M	M	C
common-name	C	–	–	–	–
country-name	M	M	M	M	C
network-address	–	–	–	–	M
numeric-user-identifier	–	M	–	–	–
organization-name	C	–	–	–	–
organizational-unit-names	C	–	–	–	–
personal-name	C	–	–	–	–
private-domain-name	C	C	C	C	C
terminal-identifier	–	–	–	–	C
terminal-identifier	–	–	–	–	C
Postal routing					
physical-delivery-service	–	–	C	C	–
physical-delivery-country-name	–	–	M	M	–
postal-code	–	–	M	M	–
Postal addressing					
extension-postal-O/R-address-components	–	–	C	–	–
extension-physical-delivery-address-components	–	–	C	–	–
local-postal-attributes	–	–	C	–	–
physical-delivery-office-name	–	–	C	–	–
physical-delivery-office-number	–	–	C	–	–
physical-delivery-organization-name	–	–	C	–	–
physical-delivery-personal-name	–	–	C	–	–
poste-office-box-address	–	–	C	–	–
poste-restante-address	–	–	C	–	–
street address	–	–	C	–	–
unformatted-postal-address	–	–	–	M	–
unique-postal-name	–	–	C	–	–
Domain-defined					
domain-defined (one or more)	C	C	–	–	C

MNEM Mnemonic
NUMR Numeric
POST Postal
TERM Terminal

F Formatted
U Unformatted
M Mandatory
C Conditional

A postal O/R address, whether formatted or unformatted, comprises the following attributes:

a) one country-name and one administration-domain-name, which together identify an ADMD;

b) conditionally, one private-domain-name, one physical-delivery-service-name, or both, which together identify the PDS by means of which the user is to be accessed;

c) one physical-delivery-country-name and one postal-code, which together identify the geographical region in which the user takes delivery of physical messages.

A formatted postal O/R address comprises, additionally, one of each postal addressing attribute (see Table 9/X.402), except unformatted-postal-address, that the PDS requires to identify the postal patron.

An unformatted postal O/R address comprises, additionally, one unformatted-postal-address attribute.

Note – The total number of characters in the values of all attributes but country-name, administration-domain-name, and physical-delivery-service-name in a postal O/R address should be small enough to permit their rendition in 6 lines of 30 characters, the size of a typical physical envelope window. The rendition algorithm is PDAU-specific but is likely to include inserting delimiters (e.g., spaces) between some attribute values.

18.5.4 *Terminal O/R address*

A **terminal O/R address** is one that identifies a user by means of the network address and, if required, the type of his terminal. It may also identify the ADMD through which that terminal is accessed. In the case of a telematic terminal, it gives the terminal's network address and possibly its terminal identifier and terminal type. In the case of a telematic terminal, it gives the terminal's network address and possibly its terminal identifier and terminal type. In the case of a telex terminal, it gives its telex number.

A terminal O/R address comprises the following attributes:

a) one network-address;

b) conditionally, one terminal-identifier;

c) conditionally, one terminal-type;

d) conditionally, both one country-name and one administration-domain-name which together identify an ADMD;

e) conditionally, one private-domain-name and, conditionally, one or more domain-defined attributes, all of which provide information additional to that which identifies the user.

The private-domain-name and the domain-defined attributes shall be present only if the country-name and administration-domain-name attributes are present.

18.6 *Conditional attributes*

The presence or absence in a particular O/R address of the attributes marked conditional in Table 10/X.402 is determined as follows.

If a user or DL is accessed through a PRMD, attributes used to route messages to the PRMD are present in the O/R address at the discretion of, and in accordance with rules established by the ADMD denoted by the country-name and administration-domain-name attributes of the O/R address. The ADMD imposes no other constraints on the attributes in the O/R address. If a user is not accessed through a PRMD, all conditional attributes except those specific to postal O/R addresses are present in an O/R address at the discretion of, and in accordance with rules established by, the ADMD denoted by the country-name and administration-domain-name attributes.

All conditional attributes specific to postal O/R addresses are present or absent in such O/R addresses so as to satisfy the postal addressing requirements of the users they identify.

19 Routing

To convey a message, probe, or report toward a user or the expansion point of a DL, an MTA must not only locate the user or DL (i.e., obtain its O/R address) but also select a route to that location.

External routing is an incremental and only loosely standardized process. Suggested below are several principles of external routing. Internal routing is outside the scope of this Recommendation.

The following principles are illustrative, not definitive:

a) In an MHS that comprises a single MD, of course, routing is not an issue.

b) A PRMD may be connected to a single, ADMD. When this is so, routing always involves the ADMD necessarily.

c) An ADMD may be connected to multiple PRMDs. When this is so, routing may be based upon conditional O/R address attributes, including but not limited to private-domain-name.

d) An MD may be directly connected to some but not all other MDs. When the O/R address identifies a MD to which no direct connection exists, routing may be based upon *bilateral agreements* with the MDs to which direct connections do exist and other local rules.

e) When the MD is directly connected to the MD identified by the O/R address, the object is typically routed to that MD directly.

f) By *bilateral agreement*, one MD might route an object to another MD for the purpose, e.g., of conversion.

g) An MD may route to a malformed O/R address provided (of course) that it contains at least the attributes required to do so.

Note – The bilateral agreements and local rules alluded to above are beyond the scope of this Recommendation and may be based upon technical, policy, economic, or other considerations.

SECTION 5 – USE OF THE DIRECTORY

20 Overview

This section describes the uses to which the MHS may put the Directory if it is present. If the Directory is unavailable to the MHS, how, if at all, the MHS performs these same tasks is a local matter.

This section covers the following topics:

a) authentication;

b) name resolution;

c) DL expansion;

d) capability assessment.

21 Authentication

A functional object may accomplish authentication using information stored in the Directory.

22 Name resolution

A functional object may accomplish name resolution using the Directory.

To obtain the O/R address(es) of a user or DL whose Directory name it possesses, an object presents that name to the Directory and requests from the object's Directory entry the following attributes:

a) *MHS O/R addresses*;

b) *MHS preferred delivery methods*

To do this successfully, the object must first authenticate itself to the Directory and have access rights to the information requested.

23 DL expansion

A functional object may accomplish DL expansion using the Directory, first verifying that the necessary submit permissions exist.

To obtain the members of a DL whose Directory name it possesses, the object presents that name to the Directory and requests from the object's Directory entry the following attributes:

a) *MHS DL members*;

b) *MHS DL submit permissions*;

c) *MHS preferred delivery methods.*

To do this successfully, the MTA must first authenticate itself to the Directory and have access rights to the information requested.

24 Capability assessment

A functional object may assess the capabilities of a user or MS using the Directory.

The following Directory attributes represent user capabilities of possible significance in message handling:

a) *MHS deliverable content length*;

b) *MHS deliverable content types*;

c) *MHS deliverable EITs*;

d) *MHS preferred delivery methods.*

The following Directory attributes represent MS capabilities of possible significance in message handling:

a) *MHS supported automatic actions*;

b) *MHS supported content types*;

c) *MHS supported optional attributes.*

To assess a particular capability of a user or MS whose Directory name it possesses, the object presents that name to the Directory and requests from the object's Directory entry the attribute associated with that capability.

To do this successfully, the MTA must first authenticate itself to the Directory and have access rights to the information requested.

SECTION 6 – OSI REALIZATION

25 Overview

This section describes how the MHS is realized by means of OSI.

This section covers the following topics:

a) application service elements;

b) application contexts.

26 Application service elements

This paragraph identifies the application service elements(ASEs) that figure in the OSI realization of message handling.

In OSI the communication capabilities of open systems are organized into groups of related capabilities called ASEs. The present clause reviews this concept from the OSI reference model, draws a distinction between *symmetric* and *asymmetric* ASEs, and introduces the ASEs defined for or supportive of Message Handling.

Note – Besides the ASEs discussed, the MHS relies upon the Directory access service element defined in Recommendation X.519. However, since that ASE does not figure in the ACs for message handling (see Recommendation X.419), it is not discussed here.

26.1 *The ASE concept*

The ASE concept is illustrated in Figure 12/X.402, which depicts two communicating open systems. Only the OSI-related portions of the open systems, called AEs, are shown. Each AE comprises a UE and one or more ASEs. A UE represents the controlling or organizing portion of an AE which defines the open system's role (e.g., that of an MTA). An ASE represents one of the communication capability sets, or services (e.g., for message submission or transfer), that the UE requires to play its role.

The relationship between two AEs in different open systems is called an application association. The ASEs in each open system communicate with their peer ASEs in the other open system via a presentation connection between them. That communication is what creates and sustains the relationship embodied in the application association. For several ASEs to be successfully combined in a single AE, they must be designed to coordinate their use of the application association.

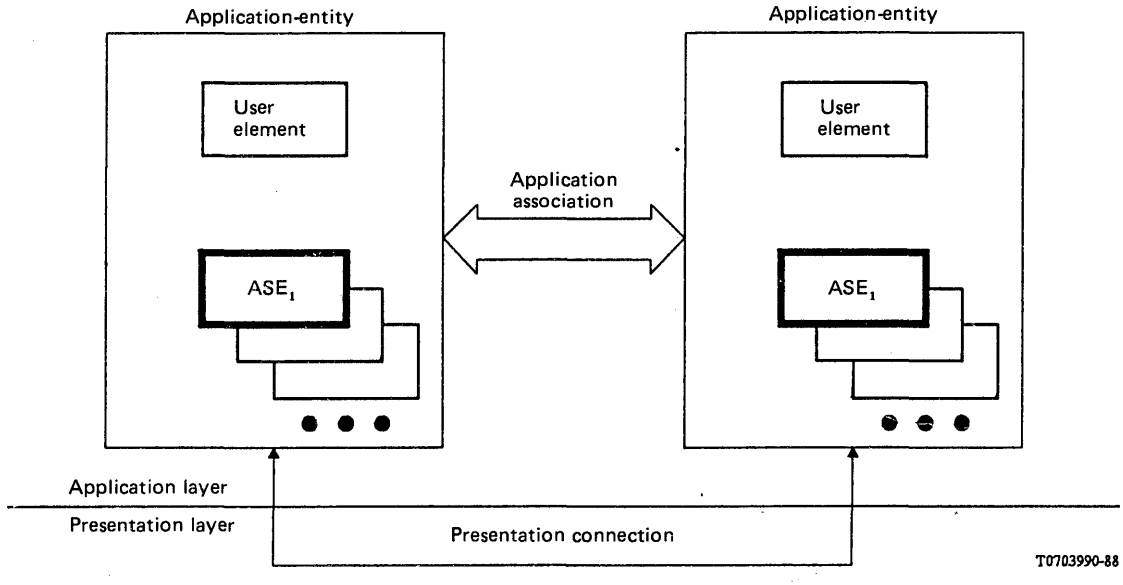


FIGURE 12/X.402

The ASE concept

An ASE plays the largely mechanical role of translating requests and responses made by its UE to and from the form dictated by the application protocol that governs the ASE's interaction with its peer ASE in the open system to which the association connects it. The ASE realizes an abstract service, or a part thereof, for purposes of OSI communication (see Recommendation X.407).

Note – Strictly speaking, an open system's role is determined by the behaviour of its application processes. In the message handling context an application process realizes a functional object of one of the types defined in § 7. A UE in turn is one part of an application process.

26.2 *Symmetric and asymmetric ASEs*

The following two kinds of ASE, illustrated in Figure 13/X.402, can be distinguished:

a) **symmetric**: Said of an ASE by means of which a UE both supplies and consumes a service. The ASE for message transfer, e.g., is symmetric because both open systems, each of which embodies an MTA, offer and may consume the service of message transfer by means of it.

b) **asymmetric**: Said of an ASE by means of which a UE supplies or consumes a service, but not both, depending upon how the ASE is configured. The ASE for message delivery, e.g., is asymmetric because only the open system embodying an MTA offers the associated service and only the other open system, which embodies a UA or MS, consumes it.

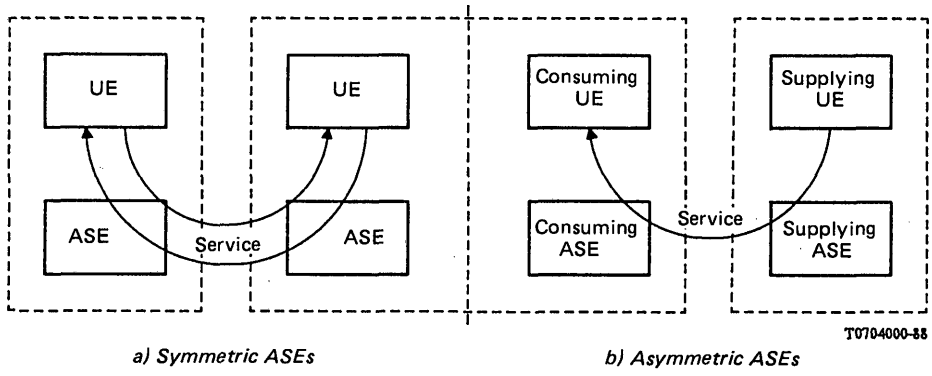


FIGURE 13/X.402

Symmetric and asymmetric ASEs

With respect to a particular asymmetric ASE, one UE supplies a service which the other consumes. The ASEs co-located with the UEs assist in the service's supply and consumption. The resulting four roles are captured in Figure 14/X.402 and in the following terminology:

a) **x-supplying UE**: An application process that supplies the service represented by asymmetric ASE x.

b) **x-supplying ASE**: An asymmetric ASE x configured for co-location with an x-supplying-UE.

c) **x-consuming UE**: An application process that consumes the service represented by asymmetric ASE x.

d) **x-consuming ASE**: An asymmetric ASE x configured for co-location with an x-consuming-UE.

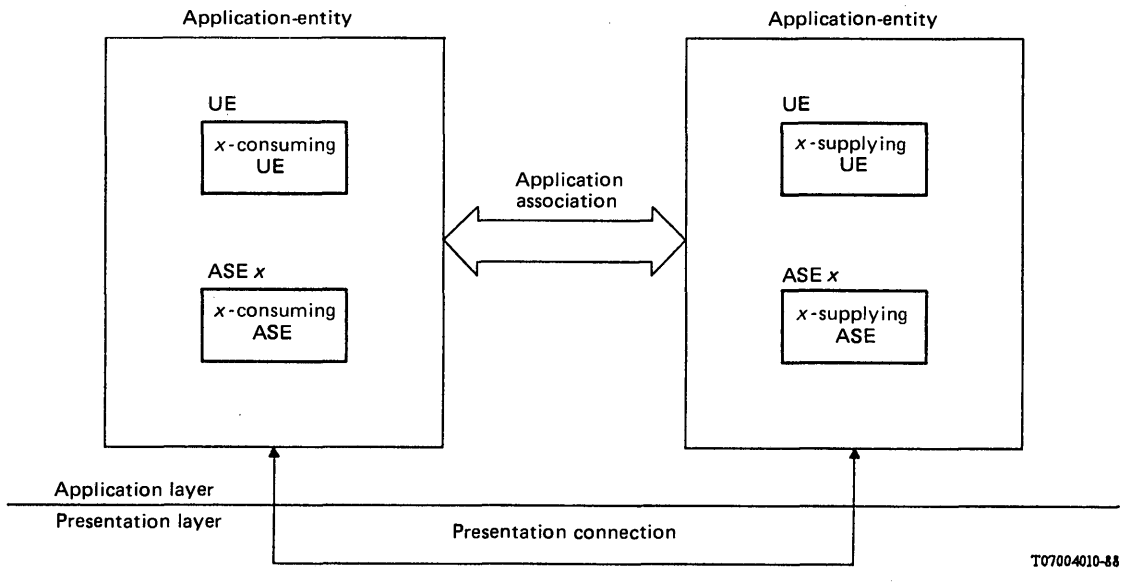


FIGURE 14/X.402

Terminology for asymmetric ASEs

As indicated, the four roles described above are defined relative to a particular ASE. When an AE comprises several asymmetric ASEs, these roles are assigned independently for each ASE. Thus, as shown in Figure 15/X.402, a single UE might serve as the consumer with respect to one ASE and as the supplier with respect to another.

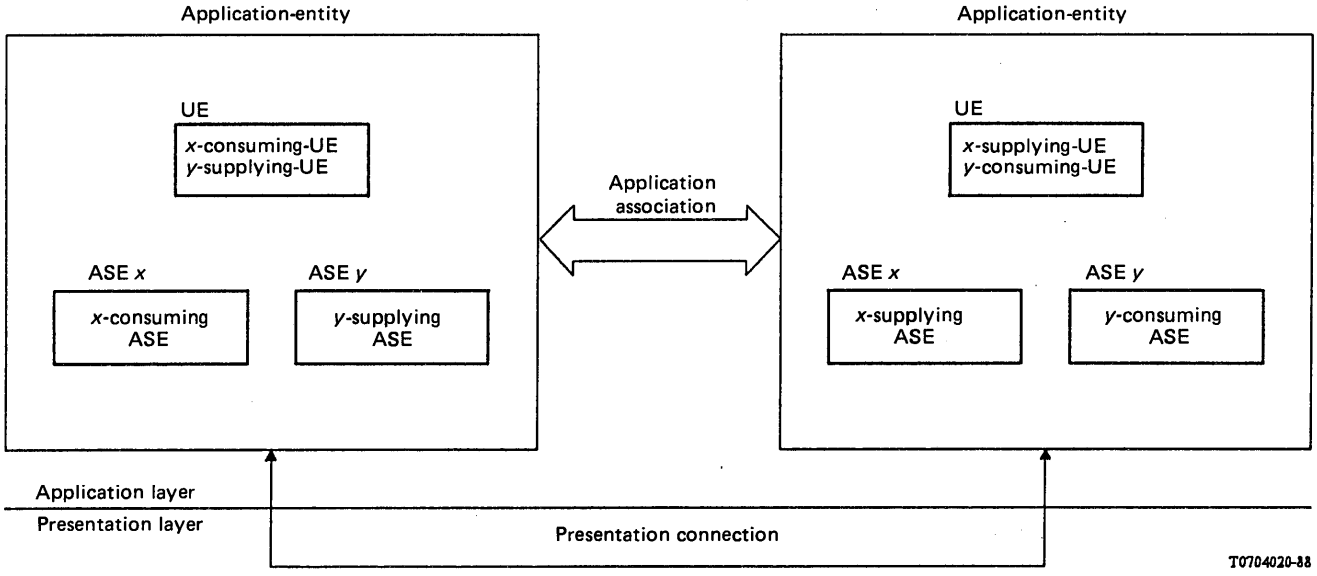


FIGURE 15/X.402

Multiple asymmetric ASEs

26.3 *Message handling ASEs*

The ASEs that provide the various message handling services are listed in the first column of Table 11/X.402. For each ASE listed, the second column indicates whether it is symmetric or asymmetric. The third column identifies the functional objects – UAs, MSs, MTAs, and AUs – that are associated with the ASE, either as consumer or as supplier.

TABLE 11/X.402

Message handling ASEs

ASE	Form	Functional objects			
		UA	MS	MTA	AU
MTSE	SY	–	–	CS	–
MSSE	ASY	C	CS	S	–
MDSE	ASY	C	C	S	–
MRSE	ASY	C	S	–	–
MASE	ASY	C	CS	S	–

SY Symmetric

ASY Asymmetric

C Consumer

S Supplier

The message handling ASEs, summarized in Table 11/X.402, are individually introduced in the clauses below. Each is defined in Recommendation X.419.

26.3.1 *Message transfer*

The message transfer service element (MTSE) is the means by which the transfer transmittal step is effected.

26.3.2 *Message submission*

The message submission service element (MSSE) is the means by which the submission transmittal step is effected.

26.3.3 *Message delivery*

The message delivery service element (MDSE) is the means by which the delivery transmittal step is effected.

26.3.4 *Message retrieval*

The message retrieval service element (MRSE) is the means by which the retrieval transmittal step is effected.

26.3.5 *Message administration*

The message administration service element (MASE) is the means by which a UA, MS, or MTA places on file with one another information that enables and controls their subsequent interaction by means of the MSSE, MDSE, MRSE, and MASE.

26.4 *Supporting ASEs*

The general-purpose ASEs upon which message handling ASEs depend are listed in the first column of Table 12/X.402. For each listed ASE, the second column indicates whether it is symmetric or asymmetric.

TABLE 12/X.402

Supporting ASEs

ASE	Form
ROSE	SY
RTSE	SY
ACSE	SY

SY Symmetric

The supporting ASEs, summarized in Table 12/X.402 are individually introduced below.

26.4.1 *Remote operations*

The remote operations service element (ROSE) is the means by which the asymmetric Message Handling ASEs structure their request-response interactions between consuming and supplying open systems.

The ROSE is defined in Recommendation X.219.

26.4.2 *Reliable transfer*

The reliable transfer service element (RTSE) is the means by which various symmetric and asymmetric message handling ASEs convey information objects – especially large ones (e.g., facsimile messages) – between open systems so as to ensure their safe-storage at their destinations.

The RTSE is defined in Recommendation X.218.

26.4.3 *Association control*

The association control service element (ACSE) is the means by which all application associations between open systems are established, released, and in other respects managed.

The ACSE is defined in Recommendation X.217.

27 Application contexts

In OSI the communication capabilities (i.e., ASEs) of two open systems are marshalled for a particular purpose by means of application contexts (ACs). An AC is a detailed specification of the use of an association between two open systems, i.e., a protocol.

An AC specifies how the association is to be established (e.g., what initialization parameters are to be exchanged), what ASEs are to engage in peer-to-peer communication over the association, what constraints (if any) are to be imposed upon their individual use of association, whether the initiator or responder is the consumer of each asymmetric ASE, and how the association is to be released (e.g., what finalization parameters are to be exchanged).

Every AC is named (by an ASN.1 object identifier). The initiator of an association indicates to the responder the AC that will govern the association's use by conveying the AC's name to it by means of the ACSE.

An AC also identifies by name (an ASN.1 object identifier) the abstract syntaxes of the APDUs that an association may carry as a result of its use by the AC's ASEs. Conventionally one assigns a name to the set of APDUs associated either with each individual ASE or with the AC as a whole. The initiator of an association indicates to the responder the one or more abstract syntaxes associated with the AC by conveying their names to it via the ACSE.

The abstract syntax of an APDU is its structure as an information object (e.g., an ASN.1 Set comprising an Integer command code and an IA5 String command argument). It is distinguished from the APDU's transfer syntax which is how the information object is represented for transmission between two open systems (e.g., one octet denoting an ASN.1 Set, followed by one octet giving the length of the Set, etc.).

The ACs by means of which the various message handling services are provided are specified in Recommendation X.419. These protocols are known as P1, P3, and P7.

Note – The nature of a message's content does not enter into the definition of message handling ACs because the content is encapsulated (as an octet string) in the protocols by means of which it is conveyed.

ANNEX A

(to Recommendation X.402)

Directory object classes and attributes

This Annex is an integral part of this Recommendation.

Several Directory object classes, attributes, and attribute syntaxes are specific to Message Handling. These are defined in the present Annex using the OBJECT-CLASS, ATTRIBUTE, and ATTRIBUTE-SYNTAX macros of Recommendation X.501, respectively.

A.1 *Object classes*

The object classes specific to message handling are those specified below.

Note – The Directory object classes described in this Annex can be combined with other object classes, eg., the ones defined in Recommendation X.521. See also Recommendation X.501, § 9, for an explanation of how Directory object classes can be combined in one Directory entry. Annex B of Recommendation X.521 gives some further information about Directory name forms and possible Directory information tree structures.

A.1.1 *MHS distribution list*

An **MHS distribution list** object is a DL. The attributes in its entry identify its common name, submit permissions, and O/R addresses and, to the extent that the relevant attributes are present, describe the DL, identify its organization, organizational units, and owner; cite related objects; and identify its deliverable content types, deliverable EITs, members, and preferred delivery methods.

```
mhs-distribution-list OBJECT-CLASS
        SUBCLASS OF top
        MUST CONTAIN {
                commonName,
                mhs-dl-submit-permissions,
                mhs-or-addresses}
        MAY CONTAIN {
                description,
                organization,
                organizationalUnitName,
                owner,
                seeAlso,
                mhs-deliverable-content-types,
                mhs-deliverable-eits,
                mhs-dl-members,
                mhs-preferred-delivery-methods }
        ::= id-oc-mhs-distribution-list
```

A.1.2 *MHS message store*

An **MHS message store** object is an AE that realizes an MS. The attributes in its entry, to the extent that they are present, describe the MS, identify its owner, and enumerate the optional attributes, automatic actions, and content types it supports.

```
mhs-message-store OBJECT-CLASS
        SUBCLASS OF applicationEntity
        MAY CONTAIN {
                description,
                owner,
                mhs-supported-optional-attributes,
                mhs-supported-automatic-actions,
                mhs-supported-content-types }
        ::= id-oc-mhs-message-store
```

A.1.3 *MHS message transfer agent*

An **MHS message transfer agent** object is an AE that implements an MTA. The attributes in its entry, to the extent that they are present, describe the MTA and identify its owner and its deliverable content length.

```
mhs-message-transfer-agent OBJECT-CLASS
        SUBCLASS OF applicationEntity
        MAY CONTAIN {
                description,
                owner,
                mhs-deliverable-content-length }
        ::= id-oc-mhs-message-transfer-agent
```

A.1.4 *MHS user*

An **MHS user** object is a generic MHS user. (The generic MHS user can have, for example, a business address, a residential address, or both.) The attributes in its entry identify the user's O/R address and, to the extent that the relevant attributes are present, identify the user's deliverable content length, content types, and EITs; its MS; and its preferred delivery methods.

```
mhs-user OBJECT-CLASS
        SUBCLASS OF top
        MUST CONTAIN {
              mhs-or-addresses }
        MAY CONTAIN {
              mhs-deliverable-content-length,
              mhs-deliverable-content-types,
              mhs-deliverable-eits,
              mhs-message-store,
              mhs-preferred-delivery-methods }
        ::= id-oc-mhs-user
```

A.1.5 *MHS user agent*

An **MHS user agent** object is an AE that realizes a UA. The attributes in its entry, to the extent that they are present, identify the UA's owner; its deliverable content length, content types, and EITs; and its O/R address.

```
mhs-user-agent OBJECT-CLASS
        SUBCLASS OF applicationEntity
        MAY CONTAIN {
              owner,
              mhs-deliverable-content-length,
              mhs-deliverable-content-types,
              mhs-deliverable-eits,
              mhs-or-addresses }
        ::= id-oc-mhs-user-agent
```

A.2 *Attributes*

The attributes specific to message handling are those specified below.

A.2.1 *MHS deliverable content length*

The **MHS deliverable content length** attribute identifies the maximum content length of the messages whose delivery a user will accept.

A value of this attribute is an Integer.

```
mhs-deliverable-content-length ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX integerSyntax
        SINGLE VALUE
        ::= id-at-mhs-deliverable-content-length
```

A.2.2 *MHS deliverable content types*

The **MHS deliverable content types** attribute identifies the content types of the messages whose delivery a user will accept.

A value of this attribute is an object identifier.

```
mhs-deliverable-content-types ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX objectIdentifierSyntax
        MULTI VALUE
        ::= id-at-mhs-deliverable-content-types
```

A.2.3 *MHS deliverable EITs*

The **MHS deliverable EITs** attribute identifies the EITs of the messages whose delivery a user will accept.

A value of this attribute is an object identifier.

```
mhs-deliverable-eits ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX objectIdentifierSyntax
        MULTI VALUE
        ::= id-at-mhs-deliverable-eits
```

A.2.4 *MHS DL members*

The **MHS DL members** attribute identifies a DL's members.

A value of this attribute is an O/R name.

```
mhs-dl-members ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX mhs-or-name-syntax
        MULTI VALUE
        ::= id-at-mhs-dl-members
```

A.2.5 *MHS DL submit permissions*

The **MHS DL submit permissions** attribute identifies the users and DLs that may submit messages to a DL.

A value of this attribute is a DL submit permission.

```
mhs-dl-submit-permissions ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX mhs-dl-submit-permission-syntax
        MULTI VALUE
        ::= id-at-mhs-dl-submit-permissions
```

A.2.6 *MHS message store*

The **MHS message store** attribute identifies a user's MS by name.

The value of this attribute is a Directory distinguished name.

```
mhs-message-store ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX distinguishedNameSyntax
        SINGLE VALUE
        ::= id-at-mhs-message-store
```

A.2.7 *MHS O/R addresses*

The **MHS O/R addresses** attribute specifies a user's or DL's O/R addresses.

A value of this attribute is an O/R address.

```
mhs-or-addresses ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX mhs-or-address-syntax
        MULTI VALUE
        ::= id-at-mhs-or-addresses
```

A.2.8 *MHS preferred delivery methods*

The **MHS preferred delivery methods** attribute identifies, in order of decreasing preference, the methods of delivery a user prefers.

A value of this attribute is a preferred delivery method.

```
mhs-preferred-delivery-methods ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX RequestedDeliveryMethod
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-at-mhs-preferred-delivery-methods
```

A.2.9 *MHS supported automatic actions*

The **MHS supported automatic actions** attribute identifies the automatic actions that an MS fully supports.

A value of this attribute is an object identifier.

```
mhs-supported-automatic-actions ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX objectIdentifierSyntax
        MULTI VALUE
        ::= id-at-mhs-supported-automatic-actions
```

A.2.10 *MHS supported content types*

The **MHS supported content types** attribute identifies the content types of the messages whose syntax and semantics a MS fully supports.

A value of this attribute is an object identifier.

```
mhs-supported-content-types ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX objectIdentifierSyntax
        MULTI VALUE
        ::= id-at-mhs-supported-content-types
```

A.2.11 *MHS supported optional attributes*

The **MHS supported optional attributes** attribute identifies the optional attributes that an MS fully supports.

A value of this attribute is an Object Identifier.

```
mhs-supported-optional-attributes ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX objectIdentifierSyntax
        MULTI VALUE
        ::= id-at-mhs-supported-optional-attributes
```

A.3 *Attribute syntaxes*

The attribute syntaxes specific to message handling are those specified below.

A.3.1 *MHS DL submit permission*

The **MHS DL submit permission** attribute syntax characterizes an attribute each of whose values is a submit permission.

```
mhs-dl-submit-permission-syntax ATTRIBUTE-SYNTAX
        SYNTAX DLSubmitPermission
        MATCHES FOR EQUALITY
        ::= id-as-mhs-dl-submit-permission

DLSubmitPermission ::= CHOICE {
        individual      [0] ORName,
        member-of-dl    [1] ORName,
        pattern-match   [2] ORNamePattern,
        member-of-group [3] Name }
```

A presented DL submit permission value shall be of type *Individual.*

A DL submit permission, depending upon its type, grants submit access to the following zero or more users and DLs:

a) *Individual*: The user or (unexpanded) DL any of whose O/R names is equal to the specified O/R name.

b) *Member-of-dl*: Each member of the DL, any of whose O/R names is equal to the specified O/R name, or of each nested DL, recursively.

c) *Pattern-match*: Each user or (unexpanded) DL any of whose O/R names matches the specified O/R name pattern.

```
ORNamePattern ::= ORName
```

d) *Member-of-group*: Each member of the group-of-names whose name is specified, or of each nested group-of-names, recursively.

A presented value is equal to a target value of this type if the two are identical, attribute by attribute. Additionally, equality may be declared under other conditions which are a local matter.

A.3.2 *MHS O/R address*

The **MHS O/R address** attribute syntax characterizes an attribute each of whose values is an O/R address.

```
mhs-or-address-syntax ATTRIBUTE-SYNTAX
        SYNTAX ORAddress
        MATCHES FOR EQUALITY
        ::= id-as-mhs-or-address
```

A presented O/R address value is equal to a target O/R address value under the conditions specified in § 18.4.

A.3.3 *MHS O/R name*

The **MHS O/R name** attribute syntax characterizes an attribute each of whose values is an O/R name.

```
mhs-or-name-syntax ATTRIBUTE-SYNTAX
        SYNTAX ORName
        MATCHES FOR EQUALITY
        ::= id-as-mhs-or-name
```

A presented O/R name value is equal to a target O/R name value if the two are identical, attribute by attribute. Additionally, equality may be declared under other conditions which are a local matter.

ANNEX B

(to Recommendation X.402)

Reference definition of object identifiers

This Annex is an integral part of this Recommendation.

This Annex defines for reference purposes various object identifiers cited in the ASN.1 module of Annex C. It uses ASN.1.

All object identifiers this Recommendation assigns are assigned in this Annex. Annex B is definitive for all but those for ASN.1 modules and MHS itself. The definitive assignments for the former occur in the modules themselves; other references to them appear in IMPORT clauses. The latter is fixed.

```
MHSObjectIdentifiers { joint-iso-ccitt
        mhs-motis(6) arch(5) modules(0) object-identifiers(0) }
        DEFINITIONS IMPLICIT TAGS ::=
        BEGIN
        -- Prologue
        -- Exports everything.

IMPORTS -- nothing -- ;

ID ::= OBJECT IDENTIFIER

-- Aspects MHS

id-mhsac ID ::= { joint-iso-ccitt mhs-motis(6) mhsac(0) }
            -- MHS Application Contexts
            -- See Recommendation X.419.
```

```
id-ipms     ID ::= { joint-iso-ccitt mhs-motis(6) ipms(1) }
            -- Interpersonal Messaging
            -- See Recommendation X.420.

id-asdc     ID ::= { joint-iso-ccitt mhs-motis(6) asdc(2) }
            -- Abstract Service Definition Conventions
            -- See Recommendation X.407.

id-mts      ID ::= { joint-iso-ccitt mhs-motis(6) mts(3) }
            -- Message Transfer System
            -- See Recommendation X.411.

id-ms       ID ::= { joint-iso-ccitt mhs-motis(6) ms(4) }
            -- Message Store
            -- See Recommendation X.413.

id-arch     ID ::= { joint-iso-ccitt mhs-motis(6) arch(5) }
            -- Overall Architecture
            -- See this Recommendation.

id-group    ID ::= { joint-iso-ccitt mhs-motis(6) group(6) }
            -- Reserved.


-- Categories

id-mod      ID ::= { id-arch 0 } -- modules, not definitive
id-oc       ID ::= { id-arch 1 } -- object classes
id-at       ID ::= { id-arch 2 } -- attribute types
id-as       ID ::= { id-arch 3 } -- attribute syntaxes


-- Modules

id-object-identifiers                ID ::= { id-mod 0 } -- not definitive
id-directory-objects-and-attributes; ID ::= { id-mod 1 }
                    -- not definitive


-- Object classes

id-oc-mhs-distribution-list          ID ::= { id-oc 0 }
id-oc-mhs-message-store              ID ::= { id-oc 1 }
id-oc-mhs-message-transfer-agent     ID ::= { id-oc 2 }
id-oc-mhs-user                       ID ::= { id-oc 3 }
id-oc-mhs-user-agent                 ID ::= { id-oc 4 }


-- Attributes

id-at-mhs-deliverable-content-legth      ID ::= { id-at 0 }
id-at-mhs-deliverable-content-types      ID ::= { id-at 1 }
id-at-mhs-deliverable-eits               ID ::= { id-at 2 }
id-at-mhs-dl-members                     ID ::= { id-at 3 }
id-at-mhs-dl-submit-permissions          ID ::= { id-at 4 }
id-at-mhs-message-store                  ID ::= { id-at 5 }
id-at-mhs-or-addresses                   ID ::= { id-at 6 }
id-at-mhs-preferred-delivery-methods     ID ::= { id-at 7 }
id-at-mhs-supported-automatic-actions    ID ::= { id-at 8 }
id-at-mhs-supported-content-types        ID ::= { id-at 9 }
id-at-mhs-supported-optional-attributes  ID ::= { id-at 10 }


-- Attribute syntaxes

id-as-mhs-dl-submit-permission       ID ::= { id-as 0 }
id-as-mhs-or-address                 ID ::= { id-as 1 }
id-as-mhs-or-name                    ID ::= { id-as 2 }

END -- of MHSObjectIdentifiers
```

ANNEX C

(to Recommendation X.402)

Reference definition of directory object classes and attributes

This Annex is an integral part of this Recommendation.

This Annex, a supplement to Annex A, defines for reference purposes the object classes, attributes, and attribute syntaxes specific to Message Handling. It uses the OBJECT-CLASS, ATTRIBUTE, and ATTRIBUTE-SYNTAX macros of Recommendation X.501.

```
MHSDirectoryObjectsAndAttributes { joint-iso-ccitt
    mhs-motis(6) arch(5) modules(0) directory(1) }
DEFINITIONS IMPLICIT TAGS ::=
BEGIN
    -- Prologue
    -- Exports everything.

IMPORTS
    -- MHS object identifiers
        id-as-mhs-dl-submit-permission, id-as-mhs-or-address,
        id-as-mhs-or-name-, id-at-mhs-deliverable-content-length,
        id-at-mhs-deliverable-content-types,
        id-at-mhs-deliverable-eits, id-at-mhs-dl-members,
        id-at-mhs-dl-submit-permissions, id-at-mhs-message-store,
        id-at-mhs-or-addresses, id-at-mhs-preferred-delivery-methods,
        id-at-mhs-supported-automatic-actions,
        id-at-mhs-supported-content-types,
        id-at-mhs-supported-optional-attributes,
        id-oc-mhs-distribution-list, id-oc-mhs-message-store,
        id-oc-mhs-message-transfer-agent,
        id-oc-mhs-user,
        id-oc-mhs-user-agent
        ----
        FROM MHSObjectIdentifiers { joint-iso-ccitt
            mhs-motis(6) arch(5) modules(0) object-identifiers(0) }
    -- MTS Abstract service
        ORAddress, ORName, RequestedDeliveryMethod
        ----
        FROM MTSAbstractService { joint-iso-ccitt
            mhs-motis(6) mts(3) modules(0) MTS-abstract-service(3) }
    -- Information framework
        ATTRIBUTE, ATTRIBUTE-SYNTAX, Name, OBJECT-CLASS
        ----
        FROM informationFramework { joint-iso-ccitt
            ds(5) modules(1) informationFramework(1) }
    -- Selected object classes
        applicationEntity
        top
        ----
        FROM SelectedObjectClasses { joint-iso-ccitt
            ds(5) modules(1) selectedObjectClasses(6) }
    -- Selected attribute types
        commonName, description, distinguishedNameSyntax,
        integerSyntax, objectidentifierSyntax, organization,
        organizationalUnitName, owner, seeAlso
        ----
        FROM SelectedAttributeTypes { joint-iso-ccitt
            ds(5) modules(1) selectedAttributeTypes(5) }
```

```
-- OBJECT CLASSES

    -- MHS distribution list
        mhs-distribution-list OBJECT-CLASS
            SUBCLASS OF top
            MUST CONTAIN {
                commonName,
                mhs-dl-submit-permissions,
                mhs-or-addresses }
            MAY CONTAIN {
                description,
                organization,
                organizationalUnitName,
                owner,
                seeAlso,
                mhs-deliverable-content-types,
                mhs-deliverable-eits,
                mhs-dl-members,
                mhs-preferred-delivery-methods }
            :: id-oc-mhs-distribution-list

    -- MHS message store
        mhs-message-store OBJECT-CLASS
            SUBCLASS OF applicationEntity
            MAY CONTAIN {
                description,
                owner,
                mhs-supported-optional-attributes,
                mhs-supported-automatic-actions,
                mhs-supported-content-types }
            ::= id-oc-mhs-message-store

    -- MHS message transfer agent
        mhs-message-transfer-agent OBJECT-CLASS
            SUBCLASS OF applicationEntity
            MAY CONTAIN {
                description,
                owner,
                mhs-deliverable-content-length }
            ::= id-oc-mhs-message-transfer-agent

    -- MHS user
        mhs-user OBJECT-CLASS
            SUBCLASS OF top
            MUST CONTAIN {
                mhs-or-addresses }
            MAY CONTAIN {
                mhs-deliverable-content-length,
                mhs-deliverable-content-types,
                mhs-deliverable-eits,
                mhs-message-store,
                mhs-preferred-delivery-methods }
            ::= id-oc-mhs-user

    -- MHS user agent
        mhs-user-agent OBJECT-CLASS
            SUBCLASS OF applicationEntity
            MAY CONTAIN {
                owner,
                mhs-deliverable-content-length,
                mhs-deliverable-content-types,
                mhs-deliverable-eits,
                mhs-or-addresses }
            ::= id-oc-mhs-user-agent
```

```
-- ATTRIBUTES

    -- MHS deliverable content length
        mhs-deliverable-content-length ATTRIBUTE
            WITH ATTRIBUTE-SYNTAX integerSyntax
            SINGLE VALUE
            ::= id-at-mhs-deliverable-content-length

    -- MHS deliverable content types
        mhs-deliverable-content-types ATTRIBUTE
            WITH ATTRIBUTE-SYNTAX objectIdentifierSyntax
            MULTI VALUE
            ::= id-at-mhs-deliverable-content-types

    -- MHS deliverable EITs
        mhs-deliverable-eits ATTRIBUTE
            WITH ATTRIBUTE-SYNTAX objectIdentifierSyntax
            MULTI VALUE
            ::= id-at-mhs-deliverable-eits

    -- MHS DL members
        mhs-dl-members ATTRIBUTE
            WITH ATTRIBUTE-SYNTAX mhs-or-name-syntax
            MULTI VALUE
            ::= id-at-mhs-dl-members

    -- MHS DL submit permissions
        mhs-dl-submit-permissions ATTRIBUTE
            WITH ATTRIBUTE-SYNTAX mhs-dl-submit-permission-syntax
            MULTI VALUE
            ::= id-at-mhs-dl-submit-permissions

    -- MHS O/R adresses
        mhs-or-addresses ATTRIBUTE
            WITH ATTRIBUTE-SYNTAX mhs-or-address-syntax
            MULTI VALUE
            ::= id-at-mhs-or-addresses

    -- MHS message store
        mhs-message-store ATTRIBUTE
            WITH ATTRIBUTE-SYNTAX distinguishedNameSyntax
            SINGLE VALUE
            ::= id-at-mhs-message-store

    -- MHS preferred delivery methods
        mhs-preferred-delivery-methods ATTRIBUTE
            WITH ATTRIBUTE-SYNTAX RequestedDeliveryMethod
            MATCHES FOR EQUALITY
            SINGLE VALUE
            ::= id-at-mhs-preferred-delivery-methods

    -- MHS supported automatic actions
        mhs-supported-automatic-actions ATTRIBUTE
            WITH ATTRIBUTE-SYNTAX objectIdentifierSyntax
            MULTI VALUE
            ::= id-at-mhs-supported-automatic-actions

    -- MHS supported content types
        mhs-suppported-content-types ATTRIBUTE
            WITH ATTRIBUTE-SYNTAX objectIdentifierSyntax
            MULTI VALUE
            ::= id-at-mhs-supported-content-types

    -- MHS supported optional attributes
        mhs-supported-optional-attributes ATTRIBUTE
            WITH ATTRIBUTE-SYNTAX objectIdentifierSyntax
            MULTI VALUE
            ::= id-at-mhs-supported-optional-attributes
```

```
-- ATTRIBUTE SYNTAXES

    -- MHS DL submit permission
        mhs-dl-submit-permission-syntax ATTRIBUTE-SYNTAX
            SYNTAX DLSubmitPermission
            MATCHES FOR EQUALITY
            ::= id-as-mhs-dl-submit-permission
        DLSubmitPermission ::= CHOICE {
            individual          [0] ORName,
            member-of-dl        [1] ORName,
            pattern-match       [2] ORNamePattern,
            member-of-group     [3] Name }
        ORNamePattern ::= ORName

    -- MHS O/R addresses
        mhs-or-address-syntax ATTRIBUTE-SYNTAX
            SYNTAX ORAddress
            MATCHES FOR EQUALITY
            ::= id-as-mhs-or-address

    -- MHS O/R name
        mhs-or-name-syntax ATTRIBUTE-SYNTAX
            SYNTAX ORName
            MATCHES FOR EQUALITY
            ::= id-as-mhs-or-name

END -- of MHSdirectory
```

ANNEX D

Security threats

This Annex is not a part of this Recommendation.

An overview of MHS security threats is provided in § 15.1 of Recommendation X.400. This considers threats as they appear in an MHS access threats, inter-message threats, and message store threats. These threats can appear in various forms as follows:

a) masquerade;

b) message sequencing;

c) modification of information;

d) denial of service;

e) leakage of information;

f) repudiation;

g) other MHS threats.

In addition, they may occur by accident or by malicious intent and may be active or passive. Attacks on the MHS will address potential weaknesses and may comprise of a number of threats. This Annex deals with individual threats and although consideration is given to a number of broad classes of threat, it is not a complete list.

Table D-1/X.402 indicates how these threats can be met using the MHS security services. The list of threats given here is indicative rather than definitive.

TABLE D-1/X.402

Use of MHS security services

Threat	Services
Masquerade	
Impersonation and misuse of the MTS	Message origin authentication Probe origin authentication Secure access management
Falsely acknowledge receipt	Proof of delivery
Falsely claim to originate a message	Message origin authentication
Impersonation of an MTA to an MTS-user	Proof of submission Report origin authentication Secure access management
Impersonation of an MTA to another MTA	Report origin authentication Secure access management
Message sequencing	
Replay of messages	Message sequence integrity
Re-ordering of messages	Message sequence integrity
Pre-play of messages	
Delay of messages	
Modification of information	
Modification of messages	Connection integrity Content integrity
Destruction of messages	Message sequence integrity
Corruption of routing and other management information	
Denial of service	
Denial of communications	
MTA flooding	
MTS flooding	
Repudiation	
Denial of origin	Non-repudiation of origin
Denial of submission	Non-repudiation of submission
Denial of delivery	Non-repudiation of delivery
Leakage of information	
Loss of confidentiality	Connection confidentiality Content confidentiality
Loss of anonymity	Message flow confidentiality
Misappropriation of messages	Secure access management
Traffic analysis	Message flow confidentiality
Other threats	
Originator not cleared for message security label	Secure access management Message security labelling
MTA/MTS-user not cleared for security context	Secure access management
Misrouting	Secure access management Message security labelling
Differing labelling policies	

D.1 *Masquerade*

Masquerade occurs when an entity successfully pretends to be a different entity and can take place in a number of ways. An unauthorized MTS-user may impersonate another to gain unauthorized access to MTS facilities or to act to the detriment of the valid user, e.g., to discard his messages. An MTS-user may impersonate another user and so falsely acknowledge receipt of a message by the "valid" recipient. A message may be put into the MTS by a user falsely claiming the identity of another user. An MTS-user, MS, or MTA may masquerade as another MTS user, MS, or MTA.

Masquerade threats include the following:

a) impersonation and misuse of the MTS;

b) falsely acknowledge receipt;

c) falsely claim to originate a message;

d) impersonation of an MTA to an MTS-user;

e) impersonation of an MTA to another MTA.

A masquerade usually consists of other forms of attack and in a secure system may involve authentication sequences from valid users, e.g., in replay or modification of messages.

D.2 *Message sequencing*

Message sequencing threats occur when part or all of a message is repeated, time-shifted, or reordered. This can be used to exploit the authentication information in a valid message and resequence or time-shift valid messages. Although it is impossible to prevent replay with the MHS security services, it can be detected and the effects of the threat eliminated.

Message sequencing threats include the following:

a) replay of messages;

b) reordering of messages;

c) pre-play of messages;

d) delay of messages.

D.3 *Modification of information*

Information for an intended recipient, routing information, and other management data may be lost or modified without detection. This could occur to any aspect of the message, e.g., its labelling, content, attributes, recipient, or originator. Corruption of routing or other management information, stored in MTAs or used by them, may cause the MTS to lose messages or otherwise operate incorrectly.

Modification of information threats include the following:

a) modification of messages;

b) destruction of messages;

c) corruption of routing and other management information.

D.4 *Denial of service*

Denial of service occurs when an entity fails to perform its function or prevents other entities from performing their functions. This may be a denial of access, a denial of communications (leading to other problems like overload), a deliberate suppression of messages to a particular recipient, or a fabrication of extra traffic. The MTS can be denied if an MTA has been caused to fail or operate incorrectly. In addition, an MTS-user may cause the MTS to deny a service to other users by flooding the service with messages which might overload the switching capability of an MTA or fill up all available message storage space.

Denial of service threats include the following:

a) denial of communications;

b) MTA failure;

c) MTS flooding.

D.5 *Repudiation*

Repudiation can occur when an MTS-user or the MTS may later deny submitting, receiving, or originating a message.

Repudiation threats include the following:

a) denial of origin;

b) denial of submission;

c) denial of delivery.

D.6 *Leakage of information*

Information may be acquired by an unauthorized party by monitoring transmissions, by unauthorized access to information stored in any MHS entity, or by masquerade. In some cases, the presence of an MTS-user on the system may be sensitive and its anonymity may have to be preserved. An MTS-user other than the intended recipient may obtain a message. This might result from impersonation and misuse of the MTS or through causing an MTA to operate incorrectly. Further details on the information flowing in an MTS may be obtained from observing the traffic.

Leakage of information threats include the following:

a) loss of confidentiality;

b) loss of anonymity;

c) misappropriation of messages;

d) traffic analysis.

D.7 *Other threats*

In a multi- or single-level secure system, a number of threats may exist that relate to security labelling, e.g., routing through a node that cannot be trusted with information of particular value, or where systems use different labelling policies. Threats may exist to the enforcement of a security policy based on logical separation using security labels. An MTS-user may originate a message and assign it a label for which it is not cleared. An MTS-user or MTA may set up or accept an association with a security context for which it does not have clearance.

Other threats include the following:

a) originator not cleared for message label (inappropriate submit);

b) MTA/MTS-user not cleared for context;

c) misrouting;

d) differing labelling policies.

ANNEX E

(to Recommendation X.402)

Provision of security services in Recommendation X.411

This Annex is an integral part of this Recommendation.

Table E-1/X.402 indicates which service elements from Recommendation X.411 may be used to support the security services described in § 10.2.

TABLE E-1/X.402

MHS security service provision

Service	MIS Arguments/services
Origin authentication security services	
Message origin authentication	Message origin authentication check Message token
Probe origin authentication	Probe origin authentication check
Report origin authentication	Report origin authentication check
Proof of submission	Proof of submission request Proof of submission
Proof of delivery	Proof of delivery request Proof of delivery
Secure access management security services	
Peer entity authentication	Initiator credentials Responder credentials
Security context	Security context
Data confidentiality security services	
Connection confidentiality	Not supported
Content confidentiality	Content confidentiality algorithm identifier Message token
Message flow confidentiality	Content type
Data integrity security services	
Connection integrity	Not supported
Content integrity	Content integrity check Message token Message origin authentication check
Message sequence integrity	Message sequence number Message token
Non-repudiation security services	
Non-repudiation of origin	Content integrity check Message token Message origin authentication check
Non-repudiation of submission	Proof of submission request Proof of submission
Non-repudiation of delivery	Proof of delivery request Proof of delivery
Message security labelling	Message security label Message token Message origin authentication check
Security management security services	
Change credentials	Change credentials
Register	Register

ANNEX F

Differences between CCITT Recommendation and ISO Standard

This Annex is not a part of this Recommendation.

This Annex lists all but the purely stylistic differences between this Recommendation and the corresponding ISO International Standard.

The following are the differences that exist:

a) The ISO International Standard corresponding to this Recommendation depicts direct connection of two PRMDs in the same country, direct connection of two PRMDs in different countries, and a single PRMD connected to two ADMDs, while this Recommendation does not. (See Figure 11/X.402.)

b) The ISO International Standard corresponding to this Recommendation does not require that ADMDs and PRMDs be hierarchically related for purposes of addressing and routing, while this Recommendation does. (See §§ 14.1.1., 14.1.2, 15 and 19.)

c) Where an O/R address attribute admits both printable and teletex strings, the ISO International Standard corresponding to this Recommendation does not require that the printable string be supplies as a minimum whenever attributes are conveyed internationally, while this Recommendation does. (See § 18.2.)

ANNEX G

Index

This Annex is not a part of this Recommendation.

This Annex indexes this Recommendation. It gives the number(s) of the section(s) on which each item in each of several categories is defined. Its coverage of each category is exhaustive.

This Annex indexes items (if any) in the following categories:

a) abbreviations;

b) terms;

c) information items;

d) ASN.1 modules;

e) ASN.1 macros;

f) ASN.1 types;

g) ASN.1 values;

h) *bilateral agreements*;

i) items **for further study**;

j) items **to be supplied (fs)**.

G.1 *Abbreviations*

G.2 *Terms*

Recommendation X.403

MESSAGE HANDLING SYSTEMS: CONFORMANCE TESTING

(Melbourne, 1988)

The CCITT,

considering

(a) the need for Message Handling Systems;

(b) the need to ensure the interoperability of Message Handling Systems;

(c) the need for conformance testing specifications for Message Handling Systems;

(d) that the X.400-Series Recommendations specify Message Handling Systems;

(e) the state-of-the-art of OSI testing methodology and notation within CCITT-ISO,

unanimously declares

(1) that this Recommendation describes the testing methodology for Message Handling Systems;

(2) that this Recommendation describes a notation used to define test specifications for Message Handling Systems;

(3) that this Recommendation describes the scope and content of CCITT Conformance Testing Specification Manuals for Message Handling Systems.

CONTENTS

0 Introduction

This Recommendation describes the test methods, test criteria and test notation to be used for the conformance testing of message handling systems based on the 1984 X.400-series of Recommendations as supplemented by the X.400-series Implementor's Guide (version 5).

1 Scope and field of application

The message handling protocols in the scope of this Recommendation are contained in the 1984 X.400-series of Recommendations together with the X.400-series Implementor's Guide (version 5).

Abstract test specifications for these are contained in the CCITT Conformance Testing Specification Manuals associated with this Recommendation:

- Conformance Testing Specification Manual for IPMS (P2)
- Conformance Testing Specification Manual for MTS (P1)
- Conformance Testing Specification Manual for RTS.

Even though these Manuals are referred to by this Recommendation they are not part of it.

While the complete and correct operation of session, transport and other lower-layer protocols is required for interworking the testing of these layers is not in the scope of this Recommendation. On the other hand, X.400 conformance tests should verify that the Reliable Transfer Server (RTS) correctly uses the layers beneath it.

The tests defined in this document apply to inter-domain working (ADMD to ADMD and ADMD to PRMD). They relate to any MTA or UA in a domain that supports communications with other domains.

Conformance testing of the semantics and syntax of the actual body part information carried in a BODY PART is beyond the scope of this document.

The purpose of this Recommendation is to minimize the time and expense that manufacturers of X.400 implementations and providers of X.400 services must incur to ensure a high degree of interoperability of their equipment. This purpose is achieved by having a set of X.400 conformance test specifications. The successful joint execution of the test specifications by two implementations can be accepted as compelling evidence of the complete and correct operation of these implementations.

The scope and intention of this Recommendation is different from other CCITT Recommendations which define communication services and protocols such as the 1984 X.400-series of Recommendations. The purpose of the latter Recommendations is to unambiguously define a system. However a Recommendation for conformance testing provides a well chosen subset of tests of the virtually infinite number of tests needed to guarantee full compliance to a protocol standard. The subset is chosen in such a way that it gives a high level of confidence that tested implementations will interwork while taking into account pragmatic considerations such as time taken to perform the tests.

Testing for conformance to functional standards is beyond the scope of this Recommendation. However it is recognized that conformance tests for functional standards can be derived from this Recommendation and the associated Test Specification Manuals.

It should be recognized that the conformance testing of message handling systems may fall within the framework of national regulations and may be subject to the testing policies of Administrations which are beyond the scope of this document.

2 References (1984 version)

Recommendation X.210 Open Systems Interconnection (OSI) Layer Service Definitions Convention.

Recommendation X.400 Message Handling Systems: System Model-Service Elements.

Recommendation X.401 Message Handling Systems: Basic service elements and optional user facilities.

Recommendation X.408 Message Handling Systems: Encoded information type conversion rules.

Recommendation X.409	Message Handling Systems: Presentation transfer syntax and notation.
Recommendation X.410	Message Handling Systems: Remote operations and reliable transfer server.
Recommendation X.411	Message Handling Systems: Message transfer layer.
Recommendation X.420	Message Handling Systems: Interpersonal messaging user agent layer.
X.400 Series	Implementor's Guide version 5.

3 Definitions

3.1 *Service convention definitions*

This Recommendation makes use of the following terms defined in Recommendation X.210, (version 1984):

a) primitive;
b) request (primitive);
c) indication (primitive);
d) response (primitive);
e) confirm (primitive).

3.2 *Message handling definitions*

This Recommendation makes use of the following terms defined in Recommendation X.400, (version 1984):

a) administration management domain;
b) interpersonal message (Recommendation X.420);
c) message;
d) message transfer (Recommendation X.411);
e) originator;
f) private management domain;
g) recipient;
h) user.

4 Abbreviations

The following abbreviations are used in this Recommendation:

ADMD	Administration Management Domain
ASP	Abstract Service Primitive
DSE	Distributed Single layer Embedded testmethod
MHS	Message Handling System
IPMS	Interpersonal Messaging System
IUT	Implementation Under Test
MPDU	Message Protocol Data Unit
MT	Message Transfer
MTA	Message Transfer Agent
MTS	Message Transfer System
P1	The Message Transfer Protocol [X.411]
P2	The Interpersonal Messaging Protocol [X.420]
PCO	Point of Control and Observation
PICS	Protocol Implementation Conformance Statement

PIXIT Protocol Implementation Extra Information for Testing

PDU Protocol data unit

PRMD Private management domain

RTS Reliable Transfer Server

SAP Service Access Point

TSP Test Suite Parameter

TTCN Tree and Tabular Combined Notation

UA User Agent.

5 Conventions

No conventions are defined for this Recommendation.

6 Overview

There are two kinds of CCITT documents concerned with X.400 Conformance testing:

(a) This CCITT Recommendation entitled "X.403 Message Handling Systems – Conformance testing";

(b) Three associated CCITT Conformance Testing Specification Manuals entitled:

- Conformance Testing Specification Manual for IPMS (P2)
- Conformance Testing Specification Manual for MTS (P1)
- Conformance Testing Specification Manual for RTS

The CCITT Recommendation is intended for a wide readership. The Manuals are intended for test implementors and contain detailed test specifications.

6.1 *The X.400 conformance testing Recommendation*

This Recommendation gives the following information:

a) Conformance requirements of X.400 implementations.

b) The testing methodology.

c) The structure of the test specifications.

d) Information to be supplied by implementors as a prerequisite to conformance testing.

e) The test notation.

f) Conformance assessment procedures.

6.2 *The X.400 conformance testing specification manuals*

Three CCITT conformance testing specification manuals contain test specifications for the IPMS (P2), MTS (P1), RTS. The test specifications are written in a notation described in general terms in § 11. The conformance testing specification manuals are referred to by this Recommendation but they are not part of it.

Since the manuals contain detailed and unambiguous test specifications, users of these manuals should be familiar with the X.400-series of Recommendations and with the testing methodology used.

7 Conformance requirements

The purpose of the test specifications referenced by this Recommendation is to define tests that will establish to a high degree of confidence that the various protocol layers of an implementation under test conform to the requirements of the X.400-series of Recommendations (1984).

7.1 A system claiming to conform to the X.400 IPM-Service has to support correctly:

- the basic IPM service elements as defined in Table 2/X.400;
- the IPM Optional User facilities defined as Essential in Tables 1/X.401 and 2/X.401 (where the categorization for origination and reception should be considered);
- the IPM Optional User facilities defined as Additional in Tables 1/X.401 and 2/X.401, which are claimed to be supported;
- the requirements related to the IPM service as defined in version 5 of the X.400-series Implementor's Guide.

7.2 A system claiming to conform to the X.400 MT-service has to support correctly:

- the basic MT service elements as defined in Table 1/X.400 related to the MTS (P1) protocol;
- the MT Optional User facilities defined as Essential in Tables 3/X.401 and 4/X.401 and related to the MTS (P1) protocol;
- the MT Optional User facilities defined as Additional in Tables 3/X.401 and 4/X.401 and related to the MTS (P1) protocol, which are claimed to be supported;
- the requirements related to the P1 MT-service as defined in version 5 of the CCITT X.400-series Implementor's Guide.

7.3 system claiming to conform to the X.400 RTS-service has to support correctly:

- the RTS-services as defined in X.410;
- the requirements related to the RTS-service as defined in version 5 of the CCITT X.400-series Implementor's Guide.

7.4 Claims of conformance of an implementation to the X.400-series of Recommendations can be tested using the conformance testing specification manuals associated with this Recommendation to ensure that:

(a) The implementation does not act or react in a way different to the one described in the Recommendations.

(b) The implementation is capable of handling protocol errors.

The reaction of an implementation on receipt of protocol errors is not defined in the X.400-series of Recommendations. For the purpose of conformance testing the minimum additional requirement is made that the implementation subsequently continues to operate normally in such cases.

The absence of a mandatory protocol element in P2 or P1 is regarded as a protocol error. It should be noted that in an implementated MHS a recipient domain may choose to deliver an incorrect MPDU. This should be considered as proprietary design by the equipment vendor, and the specific actions taken in these situations are defined by the vendor and not subject to conformance.

(c) The implementation correctly handles the requirements defined in X.400 Implementor's Guide Version 5.

Maximum lengths and maximum number of occurrences are interpreted in the following way:

- on origination: the implementation may support maximum lengths/occurrences up to but not exceeding the constraint value.
- on reception: the implementation must support the maximum lengths/occurrences of the constraints. Values above the constraints may be supported but the conformance requirements on the implementation upon reception of a length/occurrence exceeding the constraint are the same as for protocol errors.

Claims of conformance to the X.400 series of Recommendations can not be tested for those implementations for which it is not possible to perform all the required tests for features labeled mandatory, basic or essential optional.

8 Testing methodology

8.1 *Test configurations*

Two test configurations are used. The first configuration is shown in Figure 1/X.403 and is used to test IPMS (P2), MTS (P1) and RTS.

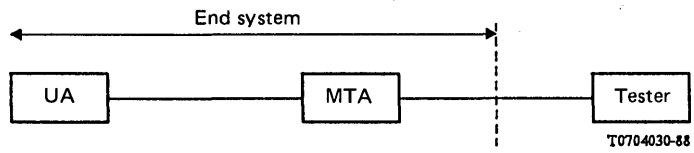


FIGURE 1/X.403

End system configuration

The second configuration is shown in Figure 2/X.403 and is used to test the relay aspects of the MTS (P1) protocol.

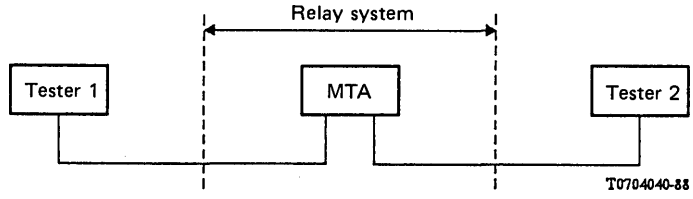


FIGURE 2/X.403

Relaying MTA test configuration

8.2 *Points of control and observation*

Test cases are described abstractly in terms of events at Points of Control and Observation (PCO) in both the tester and the Implementation Under Test (IUT). These PCOs are generally Service Access Points (SAPs) and the events are generally Abstract Service Primitives (ASPs). This does not imply that manufacturers are required to have accessible SAPs or to implement ASPs within their systems. During test execution the PCOs of an IUT may be accessed indirectly through a user interface. Where testing is performed through a user interface, the mapping of events between the SAP and the user interface is provided by the supplier of the IUT as described in § 10.2.

8.2.1 *PCOs for IPMS(P2)*

The IPMS (P2) test cases are described using the Points of Control and Observation (PCOs) shown in Figure 3/X.403.

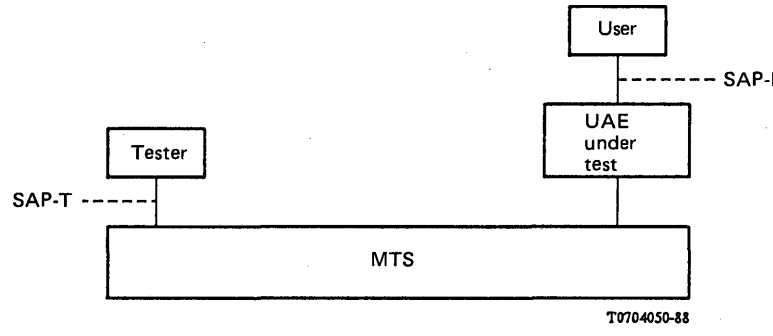


FIGURE 3/X.403

Points of control and observation for IPMS(P2)

For the tester, the Point of Control and Observation is the Service Access Point (SAP) defined at the boundary between the User Agent Layer and the Message Transfer Layer. This PCO makes use of the Message Transfer Layer Service Primitives defined in Recommendation X.411.

For the IUT, the PCO is the SAP defined at the upper boundary of the User Agent Layer. However Recommendation X.420 does not include definition of Service Primitives and it has therefore been necessary to construct hypothetical ones for sending and receiving IP-messages, in order that the test cases can be described in a formal way.

8.2.2 *PCOs for MTS(P1)*

The MTS (P1) test cases are described using the PCOs shown in Figure 4/X.403.

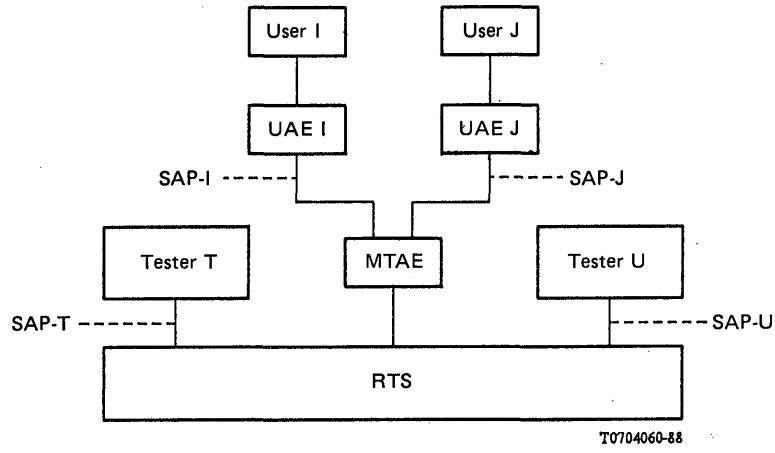


FIGURE 4/X.403

Points of control and observation for MTS(P1)

For the tester, the PCO is the SAP defined at the boundary between the MT Layer and the RTS. This PCO makes use of the RTS primitives defined in Recommendation X.410.

For the IUT, the PCO is the SAP defined at the boundary between the UA Layer and the MT Layer. This PCO makes use of the MT Service Primitives defined in Recommendation X.411.

The testing of relay functions requires more than one tester SAP. Similarly the testing of multiple destination delivery requires more than one UA on the IUT.

8.2.3 *PCOs for RTS*

The RTS test cases are described using the PCOs shown in Figure 5/X.403.

For the tester, the PCO is the SAP defined at the boundary between the RTS and the Session Layer. This PCO makes use of the Session Service Primitives defined in Recommendation X.215.

For the IUT, the PCO is the SAP defined at the upper boundary of the User Agent Layer. This PCO makes use of the same hypothetical Service Primitives defined for IPMS (P2) (§ 8.2.1).

The description of the RTS test cases includes events at a third SAP at the IUT (SAP-I) between the MT Layer and RTS. The events of this SAP are used only for clarification and it is not used as a PCO.

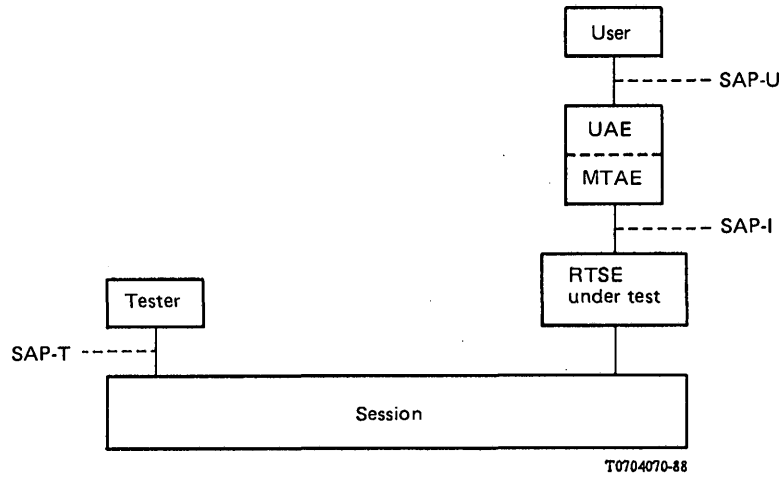


FIGURE 5/X.403

Points of control and observation for RTS

8.3 *Test design strategy*

The MHS test specifications are designed using the following concepts:

a) A test specification is defined as a test suite composed of a number of test cases as defined in § 11.1.

b) Test cases are defined in terms of:

- lower layer ASP events at the tester;
- upper layer ASP events at the IUT.

c) The test cases define the sequencing of these ASP events and the associated parameters, in particular the PDUs.

d) Test cases for valid behaviour specify ASP event sequences and PDUs that are in accordance with the X.400-series of Recommendations.

e) Test cases for invalid behaviour are characterized by:

- a correct PDU or event initiated by the tester in a protocol state where it is not permitted (an inopportune event); or
- a correct PDU incorporating an element which is syntactically correct and in range, but conflicts with the negotiated value; or
- a PDU sent by the tester which is syntactically incorrect (examples are a missing mandatory protocol element, an out-of-range value or an incorrectly encoded length indicator); or
- for RTS a lower layer ASP event issued by the tester used with parameters that are not allowed or not appropriate (example SPSN in SConnect) by X.400 restrictions.

f) The depth of testing is restricted to a reasonable number of test cases using the following principles:

1) For valid behaviour:

- if there is a small number of valid protocol element values, test all of them;
- if there is a range of values, test the bounds and a few common values;
- if there are no bounds, test an extreme value besides the common ones.

2) For invalid behaviour:

- The number of test cases for a particular type of error is reduced to one or just a few common ones.

8.3.1 *Strategy for X.409 testing*

The X.409 test cases defined in the CCITT conformance testing specification manuals associated with this Recommendation are applicable only to X.400 message handling systems. The testing of X.409 is done as part of the MTS (P1), IPMS (P2) and RTS testing. The features tested are the data types defined in X.409, the various forms of length encoding and the use of primitive and constructor data elements. To increase the likelihood that the tests can be performed, the test cases wherever possible have been defined using the protocol elements associated with mandatory service elements.

Two categories of X.409 tests are identified:

– *Decoding tests*

  These tests are constructed by identifying X.409 features to be exercised and devising sets of correctly and incorrectly encoded test PDUs containing these features. The tests are performed by transmitting the test PDUs to the IUT and observing the local reaction of the implementation and/or any PDUs returned to the tester.

– *Encoding tests*

  These tests are constructed by identifying a set of user several requests that will generate PDUs whose encoding will exercise major X.409 features. The tester must check the validity of the coding of the resulting PDUs generated by the IUT.

  The decoding tests allow the X.409 decoding features of an implementation to be fully exercised using valid and invalid test PDUs. Encoding tests only allow the valid behaviour of X.409 encoding to be checked.

8.3.2 *Strategy for IPMS(P2) testing*

Two categories of test are identified:

– IUT as originator;
– IUT as recipient.

With the IUT as originator, for each service element supported by the implementation, tests are performed by:

– invoking the service;
– the tester checking the validity of the resulting PDUs;
– where appropriate the tester returning valid and invalid response PDUs to the originator.

With the IUT as recipient, for each service element, tests are performed by:

– the tester sending valid and invalid PDUs for that service;
– observing the local reaction of the UA;
– checking the validity of any further PDUs generated by the UA.

In order to avoid unnecessary duplication of test cases, IPM service elements which are also MT service elements (for instance Delivery Notification) are listed in the MTS (P1) test suite in conjunction with the corresponding MT service elements, and not in the IPMS (P2) test suite.

It is assumed that the testing of the MT layer is done through a User Agent.

8.3.3 *Strategy for MTS(P1) testing*

When testing the operation of a MTS (P1) implementation five categories of tests are identified.

– IUT as originator;
– IUT as recipient;
– IUT as relay;
– IUT as relay recipient;
– IUT as recipient/originator.

With the IUT as originator, for each service element supported by the implementation, tests are performed by:

– invoking the service;
– checking the validity of the resulting PDUs.

With the IUT as recipient, for each service element supported by the implementation, tests are performed by:

- the tester sending valid and invalid PDUs for that service;
- observing the local reaction of the UA;
- checking the validity of any further PDUs generated by the UA.

With the IUT as relay, for each service element tests are performed by:

- the tester sending valid and invalid PDUs for relaying;
- checking the validity of the reaction of the IUT.

With the IUT as a relay recipient, for each service element tests are performed by:

- sending a set of valid and invalid PDUs destined for more than one recipient. At least one of these recipients is attached to the IUT and a further recipient is attached to a remote MTA such that the IUT has to relay the message;
- checking the validity of the reaction of the IUT as recipient;
- checking that the PDUs that are relayed are not corrupted and are modified appropriately.

With the IUT as a recipient/originator, for each service element supported by the implementation, tests are performed by:

- invoking the IUT to send a message to multiple recipients. At least one recipient will be attached to the IUT itself and a further recipient will be attached to a remote MTA;
- checking the validity of the reaction of the IUT as recipient;
- checking the validity of the PDUs transmitted by the IUT.

8.3.4 *Strategy for RTS testing*

The following testing phases are used:

a) *The connection/association establishment and negotiation phase*

The X.410 Recommendation allows different negotiable options and the negotiation phase is tested exhaustively using valid and invalid elements.

b) *The orderly release of the connection/association*

Only a few tests are required to check the correct implementation of the RTS release features.

c) *The data transfer phase with token exchange*

The data transfer tests check:

- the correct operation of data transfer using the negotiated values;
- the correct operation of token exchange;
- the correct confirmation of confirmed services;
- the correct reaction to invalid (e.g. non-negotiated) elements.

d) *Recovery*

Tests are performed to check that an IUT can perform correct recovery after:

- user aborts;
- provider aborts;
- exception reports;
- not acknowledged checkpoints.

9 Structure of test suites

The IPMS (P2) and MTS (P1) test suites have a common structure which differs from that of the RTS test suites.

9.1 *Structure of IPMS(P2) and MTS(P1) test suites*

The IPMS (P2) and MTS (P1) test suites consist of five groups of test cases:

a) *Initial tests*

The initial tests check mandatory features in a small number of test cases. They have been defined in order to check that the implementation correctly supports the main mandatory features and that it is sensible to continue with full conformance testing.

b) *X.409 tests*

The X.409 tests check the IUT's encoding and decoding of protocol elements. Decoding tests are performed by transmitting test PDUs to the IUT. Encoding tests are performed by checking PDUs received from the IUT.

c) *Protocol element tests*

Protocol element tests identify test purposes for every protocol element in the IPMS (P2)/MTS (P1) protocols. This is important in ensuring a full test coverage for the IPMS (P2)/MTS (P1) protocols. Many of these tests are necessarily performed as part of the service element tests.

d) *Service element tests*

Service element tests check the capability of the IUT to support the service elements in X.400. Some of these tests are carried out in the initial tests and the X.409 tests. Service element tests include both tests for specific service elements and tests for combinations of interdependent service elements.

e) *Additional test*

The additional test group checks features not covered in the other test groups.

As indicated in a) to e) above the number of test cases has been minimized by taking advantage of the fact that the performance of a given test case may cover more than one test purpose. Figure 6/X.403 shows how some of the test purposes identified in a particular test group may actually be achieved by test cases in another group.

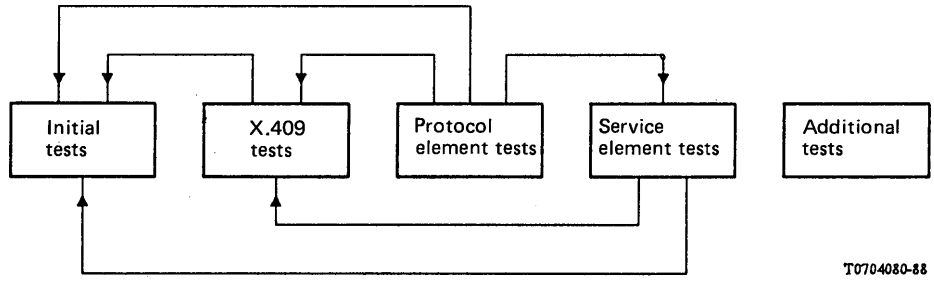


FIGURE 6/X.403

Structure of IPMS(P2) and MTS(P1) test suites

9.2 *Structure of RTS test suites*

The RTS test suite is made up of five groups of test cases:

- association establishment tests;
- association release tests;
- data transfer tests;
- association recovery tests;
- X.409 tests.

The association establishment tests check the negotiation of the connection elements.

The association release tests check the orderly release of associations.

The data transfer tests check that data is transferred correctly in accordance with the values of the connection elements negotiated during association establishment.

The association recovery tests check that the IUT can recover from breaks in connection both inside and outside activities.

The X.409 tests check the IUT's encoding and decoding of session service user data.

10 Information to be supplied by implementors

10.1 *Protocol implementation conformance statement (PICS)*

The Protocol Implementation Conformance Statement (PICS) is information supplied by an implementor that specifies the protocol features implemented in a Message Handling System.

This information is used during conformance testing:

- to check that the protocol features that have been implemented are consistent with the conformance requirements, in terms of optional and mandatory features, of the X.400-series of Recommendations;
- to select the originator tests to be executed. Recipient and relay tests will be performed to check the behaviour of the system even when it is requested to handle features that it does not implement.

PICS *proformas* for IPMS (P2), MTS (P1) and RTS are shown in Annexes B, C and D. These *proformas* specify the information to be supplied by an implementor concerning:

- the services that are supported for origination, reception and relay functions;
- the protocol features that have been implemented in order to support the services.

The IPMS (P2) PICS explicitly includes the MTS (P1) service elements made available by the IPMS (P2). In order to avoid duplication with the MTS (P1) test suite, tests for such MTS (P1) service elements are not contained in the IPMS (P2) test suite. Where the testing of MTS (P1) is not performed using a UA, MTS (P1) tests may need to be repeated using a UA in order to ensure conformance to the IPMS (P2).

10.2 *Protocol implementation extra information for testing (PIXIT)*

The Protocol Implementation extra Information for Testing (PIXIT) is supplied by an implementor specifying information needed by a tester to execute a test suite.

The IPMS (P2), MTS (P1) and RTS test suites define the behaviour of the implementation in terms of abstract service primitives. In order to invoke and observe this behaviour during test execution the test operator must know how (if at all) these abstract service primitives can be invoked or observed at the real accessible user interface.

The IPMS (P2), MTS (P1) and RTS PIXIT *proformas* will list all the IUT upper layer abstract service primitives used in the test definitions and will ask the implementor to specify how these primitives can be invoked or observed (if at all).

11 Test notation

11.1 *Definitions*

The notation used to define the MHS test specifications makes use of the following definitions:

a) **test suite**

A set of test cases, possibly combined into nested test groups, necessary to perform conformance testing of an implementation.

The test suites do not imply an order of execution.

b) **test group**

A set of related test cases. Test groups may be nested to provide a logical structuring of test cases.

c) **test case**

Specifies the sequences of test events required to achieve the purpose of the test and to assign a verdict "pass", "fail" or "inconclusive".

d) **test event**

An indivisible unit of test specification at the level of abstraction of the specification (e.g. sending or receiving a single PDU).

e) **user**

A user-interface process or a computer application which makes use of an MHS.

11.2 *Notation*

The Conformance Test Suites for Message Handling Systems use the Tree and Tabular Combined Notation as described in Annexe A of this Recommendation.

Each test suite specification is defined in six sections:

1) *Introduction*

   This contains an overview describing the scope of the tests and the structure of the test suite.

2) *Summary of test cases*

   This is a list of all tests giving the test identifier, the test reference and a short title for each test case in the test suite.

3) *Declarations part*

   Declares the names and types of all items to be used in defining the test cases.

4) *Dynamic part*

   This is the main body of the test suite and defines test cases in terms of trees of behaviour.

5) *Constraints part*

   Specifies the values of the ASPs and PDUs used in the dynamic part.

6) *Cross references*

   Provides an index to all values used in the main body of the test suite.

12 Conformance assessment procedures (see Figure 7/X.403)

This Recommendation deals only with abstract test specifications for Message Handling Systems. It does not deal with the realization of these test specifications nor with their execution. This clause in the Recommendation is purely for information purposes to describe in general terms how real testing may be done.

12.1 *Overview of the procedure*

The procedures needed to assess the conformance of an implementation include:

- the completion of the PICS and PIXIT proformas by the supplier of the implementation;
- the assessment of these documents;
- the selection and execution of test cases;
- the analysis of the results and the production of test reports.

12.2 *Analysis of PICS*

The first phase in conformance assessment is to ensure that the features claimed to be supported by an IUT comply with appropriate conformance requirements. The conformance requirements for IPMS (P2), MTS (P1) and RTS implementations are defined in § 7 of this document. This check is performed by analysing the information in the PICS documents.

12.3 *Test case selection*

The tests to be performed are selected primarily on the basis of information in the PICS. For every supported feature claimed in the PICS the corresponding test cases in the test suites are selected and executed to check the correct implementation of these features under an extensive range of valid and invalid conditions.

For non-supported features, some recipient test cases shall be executed to explore the response of the IUT. Since in general the X.400 (1984) Series of Recommendations do not define the expected behaviour in these situations, these tests can be "passed" with almost any behaviour apart from catastrophic failure by the IUT.

Information in the PIXIT may also provide some constraints on the test cases that can be executed.

12.4 *Execution of tests*

It is recommended that the testing of Message Handling Systems should be done in the order of RTS, then MTS (P1) and then IPMS (P2) testing.

However the order of test cases in the test suites does not imply an order of execution. Apart from the general recommendation that for IPMS (P2)/MTS (P1) the Initial Test Group should be executed first, the order of execution of tests can be determined by the test operators taking into account their test environment and test tools.

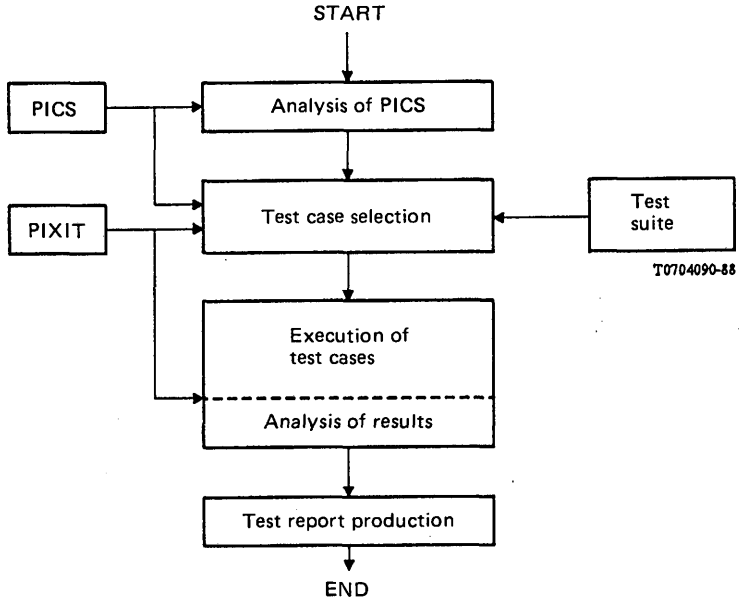


FIGURE 7/X.403

The Conformance assessment procedure

ANNEX A

(to Recommendation X. 403)

Test notation

A.1 *Introduction*

This annex is an integral part of this Recommendation and describes the notation used in the test suite manuals.

The test notation described here is based on the test notation called Tree and Tabular Combined Notation (TTCN) that has been developed jointly by ISO and CCITT.

The notation described in this Recommendation is derived from an early form of TTCN and has been developed specifically for use in the MHS conformance testing specifications.

Each of the MHS test suites is specified in five parts:

- Declaration part;
- Dynamic part;
- Constraints part;
- Test case identification;
- Cross-references.

A.2 *Declaration part*

The Declaration Part declares the environment and objects used in the test suites and is composed of 7 sections:

- Test configurations;
- Test suite parameters (TSPs);
- Service access points (SAPs);
- Abstract service primitives (ASPs);
- Protocol data units (PDUs);
- Timers;
- Abbreviations.

A.2.1 *Test configurations*

The points of control and observation are declared in this section.

A.2.2 *Test suite parameters*

Every MHS Test Suite has a set of parameters whose values are fixed prior to testing and which are used to define a specific testing environment.

TSPs are declared in tabular form as shown in Figure A-1/X.403.

TEST SUITE PARAMETERS		
NAME	TYPE and RESTRICTIONS	COMMENTS

FIGURE A-1/X.403

Test suite parameters

By convention the name of each Test Suite Parameter in the MHS test suites is of the form:

TSP_ <name>

A.2.3 *Service access points* (SAPs)

Service Access Points are used as points of control and observation in the MHS Test Suites and are declared in tabular form as shown in Figure A-2/X.403.

SAPs	
NAME	ROLE

FIGURE A-2/X.403

Service access points

By convention the name of a SAP in the MHS Test Suites is generally one capital letter, such as T, U, V (for tester SAPs) or I, J, K (for IUT SAPs).

A.2.4 *Abstract service primitives*

Each ASP type and its associated parameters used in a test suite is declared in tabular form as shown in Figure A-3/X.403.

ASP:	SAP:	COMMENTS:	
NAME	RANGE OF VALUES OR TYPE	COMMENTS	C/M

FIGURE A-3/X.403

Abstract service primitives

The name of the ASP is specified in the "ASP" field and is derived from the corresponding name in the X.400-series of Recommendations. The SAP at which the ASP occurs is specified in the "SAP" field. The parameters of the ASP are declared in the "NAME" column together with information in "RANGE OF VALUES OR TYPE", "COMMENTS" and "Conditional/Mandatory" columns.

Since there are no IPMS (P2) ASPs defined in the Recommendations, in order to describe conformance tests it has been necessary to construct hypothetical ASPs at the upper boundary of the User Agent Layer. This does not imply, however that manufacturers are required to implement these ASPs within their systems. It serves only to formalize the requirements for observation and invocation of IPMS service elements by the use of these new ASPs. The relation between IPMS service elements and the actual behaviour of the IUT has to be specified in the implementation-dependent PIXIT.

A.2.5 *Protocol data units*

The PDU types used in test suites are declared in the form of tables as shown in Figure A-4/X.403. These PDUs are not defined explicitly in the test suite, but are given a precise reference to the full definition in the X.400 Recommendations, in the type name section of the table.

DATA TYPE DECLARATION
COMMENTS:
TYPE NAME:

FIGURE A-4/X.403

Protocol data units

A.2.6 *Timers*

This section declares the timers to be used. Timer values are expressions in terms of Test Suite Parameters, and are fixed for the whole test suite. Timer values are declared in tabular form as shown in Figure A-5/X.403.

TIMER DECLARATION		
TIMER NAME	VALUE	COMMENT

FIGURE A-5/X.403

Timers

A.2.7 *Abbreviations*

Abbreviations used in the Test Suite are defined in the form of a table as shown in Figure A-6/X.403.

ABBREVIATIONS		
ABBREVIATION	FULL NAME	COMMENT

FIGURE A-6/X.403

Abbreviations

A.3 *Dynamic part*

The Dynamic Part defines the test cases of a test suite in terms of trees of behaviour.

Sections A.3.1 and A.3.2 describe generally how trees of behaviour are defined.

Section A.3.3 describes the content and use of Defaults Library.

Section A.3.4 describes the content and use of Test Step Library.

Section A.3.5 describes how each test case in the main body of a test suite is specified.

A.3.1 *Proforma table for test behaviours* (see Figure A-7/X.403)

<title> BEHAVIOUR				
IDENTIFIER: <used only for libraries>				
COMMENTS: <used only for libraries>				
DEFAULTS:				
BEHAVIOUR DESCRIPTION	LABEL	CONSTRAINTS REFERENCE	COMMENTS	RESULTS
Extended Comments: <optional>				

FIGURE A-7/X.403

Behaviour description

<title> BEHAVIOUR

Title of the behaviour: DEFAULT for the Default Library; DYNAMIC for the Test Step Library and test cases.

IDENTIFIER

This provides a unique identifier for the behaviour description.

DEFAULTS

This lists the identifiers of default behaviour descriptions which are to be used in conjunction with the Dynamic behaviour shown in the "BEHAVIOUR DESCRIPTION" part.

BEHAVIOUR DESCRIPTION

Test behaviour is defined using a tree notation as described in A.3.2.

LABEL

The LABELS column may be used to identify events. Branches between events (i.e. "GO TO") are specified by "→ Label" in the behaviour tree.

CONSTRAINTS REFERENCE

For each ASP event of a behaviour tree line, this column gives the reference to the specific ASP value defined in the Constraints Part.

COMMENTS

This column provides comments which ease understanding of the events. Additional comments may be given in the "Extended Comments" area. This column can also be used to identify test PDUs associated with test events.

RESULT

This column indicates which test events generate test verdicts. Values of test verdicts are:

- pass: no misbehaviour of the IUT is detected;
- fail: misbehaviour of the IUT is detected;
- inconclusive: the observed behaviour does not allow the assignment of a pass or fail verdict.

A.3.2 *Tree notation for test behaviours*

Trees of behaviour are defined in terms of events which are generally of the form:

<SAP>!<event>

or of the form

<SAP>?<event>

The <SAP> is the point of control and observation at which the <event> occurs. The SAPs used are those declared in the Declaration Part.

The "!" symbol indicates that the event is sent from the SAP and "?" indicates that the event is received at the SAP.

The <event> can be:

- an ASP event;
- a timer event;
- an OTHERWISE pseudo-event.

A.3.2.1 *Single ASP events*

If the <event> is an ASP event then the names for the ASPs are those specified in the Declaration Part (the value is specified as a reference in the CONSTRAINTS REFERENCE column).

Example line for an ASP event:

I?DELind

This means that a Deliver Indication is received at the IUT's SAP I.

A.3.2.2 *Single timer events*

If <event> is a timer event then it is of the form:

<operation> <parameters>

The "start" operation can take one of two forms:

Start <timer type>

Start (<timer type>, <timer id>)

Where <timer type> is defined in the Declaration Part and has a fixed value associated with it defined in terms of TSPs. The <timer id> allows a name to be attached to an instance of a timer type.

The other operations are:

- Cancel: cancels a running or suspended timer;
- Suspend: suspends a running timer;
- Resume: resumes a suspended timer;
- Timeout: expiration of a running timer;

These operations take one of two forms:

<operation> <timer type>

<operation> <timer id>

where <operation> denotes the operation. When the timer was started using the form "Start <timer type>", the form "<operation> <timer type> must be used; when the timer was started using the form "Start <timer id>", the form "<operation> <timer id>" must be used.

Example:

I!Start T/I-timer_1

means that at the IUT's SAP I the T/I-timer_1 (e.g. for a transmission time for a UAPDU to be transferred from the tester to the IUT's user) is started.

I?Timeout T/I-timer_1

means that at the IUT's SAP I the timeout of the above timer is received.

A.3.2.3 *Single OTHERWISE events*

If <event> is the OTHERWISE pseudo-event, this indicates an unspecified event.

Example:

T?OTHERWISE

Means that at the tester's SAP T an unspecified event is received.

A.3.2.4 *Trees of behaviour*

Trees of behaviour combine events in two ways:

- as sequences of events
- as alternative events

The two combination kinds are distinguished by indented and vertical alignment respectively.

Example of a sequence of events:

I!SUBreq
 I?SUBcon
  T?TRNind

This means that first at the SAP I a Submission Request is sent, then at the same SAP a Submission Confirmation is received, after which a Transfer Indication is received at the tester's SAP T.

Example of alternative events:

T?DELind
T?Timeout I/T-timer

This means that at SAP T either a Deliver Indication is received or alternatively the timeout is received there.

To build up a complex behaviour tree, the two kinds of combination can be mixed.

Example:

I!SUBreq
 I?SUBcon
 T?TRNind } alternative events
T?DISind

This means that after sending a Submission Request at I, either a Submission Confirmation is received at I, followed by the receipt of a Transfer Indication at T, or a Disconnect Indication is received at T.

A.3.3 *Defaults library*

General default behaviours which are used by several test cases are defined in the Defaults Library using the format shown in Figure A-8/X.403. The name of the default is of the form:

LIB_ <name>

or

LIB_ <name> [X]

where X is a place holder which is replaced by an actual SAP when applying the default element in a particular Test Case.

Note – Where particular default behaviour applies to a single test case only the behaviour table is associated with that test case and the identifier is not prefixed with "LIB_".

DEFAULT BEHAVIOUR				
DEFAULT IDENTIFIER: LIB_				
COMMENTS:				
DEFAULTS:				
BEHAVIOUR DESCRIPTION	LABEL	CONSTRAINTS REFERENCE	COMMENTS	RESULTS
Extended Comments: <optional>				

FIGURE A-8/X.403

Default behaviour

A.3.4 *Test step library* (see Figure A-9/X.403)

Where a sequence of test steps is of use in several test cases they can be included in the Test Step Library and given a name of the form:

LIB_ <name>

Note – Where a test step applies to a single test case the behaviour table is associated with that test case and the identifier is not prefixed with "LIB_".

A.3.5 *Test case* (see Figure A-10/X.403)

Each test case in the main body of the test suite is described in terms of three headings a)-c), and a behaviour tree d):

a) *Test reference and test identifier*

These elements give a unique reference and identifier for each test case and are described fully in § A.5.

b) *Summary*

A brief overview of the purpose of the test is provided.

c) *Test description* (optional)

This provides an informal description of the actions and events that should take place during the test and an informal verdict criteria.

d) *Behaviour tree*

Dynamic behaviour is described using the tree notation defined in § A.3.2.

DYNAMIC BEHAVIOUR				
TEST STEP IDENTIFIER: LIB_				
COMMENTS:				
DEFAULTS:				
BEHAVIOUR DESCRIPTION	LABEL	CONSTRAINTS REFERENCE	COMMENTS	RESULTS
Extended Comments: <optional>				

FIGURE A-9/X.403

Test step behaviour

DYNAMIC BEHAVIOUR				
DEFAULTS: (see note 1)				
BEHAVIOUR DESCRIPTION	LABEL	CONSTRAINTS REFERENCE	COMMENTS	RESULTS
(see Note 2)				(see Notes 3 and 4)
Extended Comments: <optional>				

Note 1 – In this field all Default Library Identifiers used are inserted. Where necessary, the SAP at which they are applied is also identified. If for example the field contains the entry:

LIB_unexpected [T]

it means that the subtree associated with this Default Behaviour is considered to be associated with the SAP T.

Note 2 – Test Step Library behaviour is included in the behaviour tree using the following notation:

+ <Test Step Library Identifier>.

Note 3 – The behaviour tree of every Test Case provides the verdicts pass, fail, and where appropriate inconclusive.

Note 4 – When using Default Library elements it is possible that some of the verdict alternatives are "hidden" in the Default Library element.

FIGURE A-10/X.403

Test case behaviour

A.4 *Constraints part* (see Figure A-11/X.403)

The Constraints Part of a Test Suite specifies the values and their encoding of all instances of ASPs, Test PDUs, Base PDUs and Library Components. The Constraints Parts is divided into the following sections:

- Introduction to Constraints Part;
- ASP Constraints;
- Test PDU Constraints;
- Base PDU Constraints;
- Components Library.

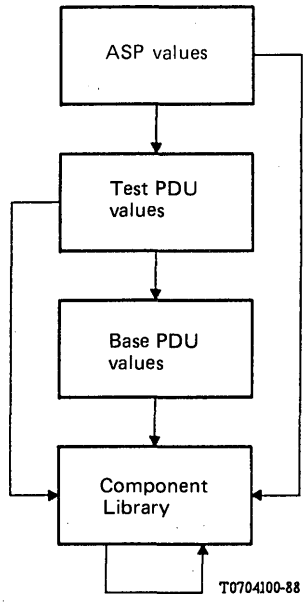


FIGURE A-11/X.403

The structure of the Constraints Part

A.4.1 *ASP constraints*

Values of ASPs are defined as specific instances of a generic ASP.

A.4.1.1 *Specification of a "Generic" ASP*

A generic ASP is defined using the format shown in Figure A-12/X.403.

The "FIELDS" column is used to list all the parameters of the ASP.

The "VALUE or REFERENCE" column is used to specify a value for each parameter and this can be done in four ways:

a) as a reference which can be a TSP name or a library component name;

b) as an explicit value;

c) as "-" to indicate that this parameter may be omitted in specific instances of this ASP;

d) as "?" to indicate that for "request" ASP's this parameter must have a value defined in a specific instance if it is a component of interest.

<ASP name>	<abbreviated name>	GENERIC
FIELDS	VALUE OR REFERENCE	COMMENT
<field_1>	<TSP instant>	M
<field_2>	<Lib component>	M
<field_3>	?	C
<field_4>	–	not applicable
<field_5>	–	C

FIGURE A-12/X.403

Generic ASP specification

A.4.1.2 *Specification of ASP instances*

Specific values of ASPs are defined using the tabular format shown in Figure A-13/X.403.

<ASP name>	<generic abbreviated name>	INSTANCES
INSTANCE NAME	MODIFIED PARAMETER	VALUE or REFERENCE
<instance_1>	<field_3>	<test_pdu_1>
<instance_2>	<field_3> <field_6>	<test_pdu_2> <Lib.component>
<instance_3>	<field_3> <field_7> <field_8>	<test_pdu_3> <TSP instant> <given value>

FIGURE A-13/X.403

Specific ASP value

The "INSTANCE NAME" column is used to identify specific instances of the ASP used in the test suite.

The "MODIFIED PARAMETER" column identifies, for "request" ASP's those parameters whose values are to be modified from the generic ASP specification, and for "notification" ASP's those parameters whose values are to be checked.

The "VALUE or REFERENCE" column can contain either specific values or references to library components ASPs or test PDUs.

A.4.2 *Specifying PDU values*

The MHS test suite contains a large number of test PDU values. Each PDU is defined in terms of modifications to one of the small number of "base" PDUs.

For convenience commonly used PDU components are defined in a library and are referenced by test PDUs and base PDUs.

A.4.3 *Base PDUs*

A.4.3.1 *Base PDU specification*

Base PDUs are not themselves used as test PDUs but they serve as a basis from which to derive the test PDUs. Usually only a few Base PDUs need to be specified.

The name of a Base PDU is of the form:

BASE_ <PDUtypename> _ <number>

Example of a Base PDU:

DESCRIPTION	VALUE or REFERENCE	COMMENT
BASE_IM-UAPDU_1		
SEQUENCE {		
Heading	[BASE_IM_UAPDU_1_Heading]	
Body	[BASE_IM-UAPDU_1_Body]	
}		
BASE_IM-UAPDU_1_Heading		
SET {		
IPMessageID	[L_IPMessageID_20]	
}		
BASE_IM-UAPDU_1_Body		
SEQUENCE OF {		
BodyPart	[L_BodyPart_20]	
}		

The value or value reference of each element of the structure is specified within square brackets ("[" and "]") under the VALUE or REFERENCE heading.

When specifying the encoding of a PDU for encoding/decoding tests,two additional columns are used to specify the ID Code [ID] and Length Indicator [LI] of each element of the PDU. The format for doing this is shown in the example below.

DESCRIPTION	ID	LI	VALUE or REF	COMMENT
L_Phantasy_2				
SEQUENCE {	['AO'H]	[LI]		IA5Text
SET {	['31'H]	[LI]		
repertoire INTEGER	['80'H]	[LI]	[5]	ia5
}				
IA5String	['36'H]	['8106'H]		constructor
	['04'H]	['01'H]	["1"]	
	['04'H]	['02'H]	["23"]	
}				

The values of ID and LI can be specified explicitly to allow invalid and various forms of valid codings to be defined. The mnemonic "LI" is used to indicate that any valid encoding of length is allowed.

A.4.3.2 *Identifying the components to be modified*

A component which is to be replaced in a PDU is identified by a path through the declaration of the PDU. The path is written as a list of elements, each separated from the next by a ".". The elements in the list can be labels which appear in a BASE PDU, components which appear in the left-hand side of a labeled declaration, or components which appear in the left-hand side of the expansion of a library reference in the right-hand side of a declaration.

For example, consider the following definitions:

```
Instance_
   SET{
      a                 [value]
      b                 [L_Component_1]
   }

L_Component_1
   SET{
      c                 [value]
      d                 [L_Component_2]
   }

L_Component_2
   SEQUENCE
      e                 [value]
   }
```

Note – L_Component_1 is in the Component Library.

In order to reference "a", the path would be instance_1.a.

In order to reference "e", the path would be instance_1.b.d.e.

A.4.4 *Test PDUs*

Test PDUs are defined in terms of operations on Base PDU's. These operations refer to Library Components, TSPs or specific values.

There are two kinds of test PDU:

– PDUs sent by the tester (IUT as recipient)

  By convention the names of these PDUs are of the form

  <PDU name>_x_<number>

  where x is the number of the base PDU from which the test PDU is derived.

– PDUs received by the tester (IUT as originator)

  By convention the names of these PDUs are of the form

  <PDU name>_0<number>

  where "0" indicates that these test PDUs are not derived from a base PDU.

A.4.4.1 *Test PDUs sent by the tester*

A test PDU sent by the tester to the IUT is normally constructed from a Base PDU by means of the REPLACE operation.

The specification has the form:

DESCRIPTION	VALUE or REFERENCE	COMMENT
<test PDU to be specified>		
<base PDU to be used> REPLACE <base PDU part> BY <partial ASN.1 tree>	[<value>]	

For the conventions of value assignments see § A.4.6.

Example:

DESCRIPTION	VALUE or REFERENCE	COMMENT
IM-UAPDU-1-18		

BASE_IM-UAPDU_1
REPLACE
  BASE_IM-UAPDU. Heading
BY
  SET {
  IPMessageID
  originator ORDescriptor
}
	
	[L_IPMessageID_7]
	[L_ORDescriptor_11]
		
		Library Components

To construct invalid components in test PDUs to be sent by the tester, the abstract REDEFINE operation is sometimes used. It is used together with the REPLACE operation in the following form:

DESCRIPTION	VALUE or REFERENCE	COMMENT
<test PDU to be specified>		
REDEFINE <Type to be redefined> ::= <new definition> <base PDU to be used> REPLACE <base PDU part> BY <partial ASN.1 tree>	 [<value>]	

The scope of the newly defined type is restricted to the PD definition containing the REDEFINE operation.

Note – That if the <value> is a reference to an element defined elsewhere (i.e. a TSP or a Library Component), then the new type definition does not affect the referenced element itself but only its usage in the actual PDU.

Example:

```
IM_UAPDU_1_3

REDEFINE
ORName :: = [APPLICATION 1] IMPLICIT SEQUENCE {
                                P1.StandardAttributeList
                                P2.DomainDefinedAttributeList      OPTIONAL
BASE_IM_UAPDU_1
REPLACE
   BASE_IM_UAPDU_1_Heading
BY
   SET {
       IPMessageID                                          [L_IPMessageID_1]
       originator ORDescriptor                              [L_ORDescriptor_1]
}
```

The error to be constructed here is the wrong tag of the ORName type (the correct tag would be [APPLICATION 0]). The scope of the erroneous type-definition constructed by "REDEFINE" is restricted to all occurrences of ORName in the definition of IM_UAPDU_1_3. This means that L_ORDescriptor_1 is used here with the modified ORName type, whereas the usage of this library component in other PDUs or components remains unaltered.

A.4.4.2 *Test PDUs received by the tester*

For received PDUs normally only a portion of the PDU relates to the purpose of the test.

A component of interest is identified and its value assigned using the techniques described in § A.4.3.

The specification scheme has the following form:

DESCRIPTION	VALUE or REFERENCE	COMMENT
<Test PDU to be specified>		
Partial definition – Components of interest		
<Test PDU part>	[<value>]	

Example:

DESCRIPTION		VALUE or REFERENCE	COMMENT
SR-UAPDU-O-95			
Partial definition – Components of interest			
SR-UAPDU.CHOICE.non-Receipt			
reason	INTEGER	[1]	autoforward
comments	PrintableString	["on holiday"]	
returned	IM-UAPDU	L_IMUAPDU_2]	
SR-UAPDU.report			
IPMessageID		[L_IPMessageID_15]	

A.4.5 *Component library*

Components of PDUs are defined in the library and are referenced in Base PDU specifications, Test PDU specifications and by other library components.

The name of a Library Component is of the form:

L_ <ASN.1 type name> _ <number>

and is specified using the techniques described in § A.4.3

Example:

DESCRIPTION	VALUE or REFERENCE	COMMENT
L_Phantasy_1		
SEQUENCE {		
SET {		
SET {		
ContentType	[2]	p2
originator		
PI.ORName	[TSP_OrName_1]	Test Suite Parameter
original		
SET {BIT STRING}	[{'20H'}]	IA5 Text
DeliveryFlags	['40H']	Conversion Prohibited
ThisRecipient		
PI.ORName	[TSP_ORName_1]	
submission TIME	[TPS_UTCTime_1]	
}		
}		
IM-UAPDU	[L_IM-UAPDU_1]	Library Cpt
}		

A.4.6 *Value conventions*

The following conventions are used when defining values or value references for PDU components.

Value references identify components defined either within the Component Library or within the Test Suite Parameters section. CharacterString Values can specified within double-quotes (e.g. "abc"); Bit String Values are specified within single-quotes (e.g. '0A'H or '0001'B; hexadecimal or binary notation); Integer Values are specified as numeric characters (e.g. 2); sets and sequences of values are specified within curly brackets separated by comma (e.g. {"abc", 'OA'H}).

For PDU's sent by the tester:

[?] indicates that the value has no influence on the test and may be anything that is legal according to the relevant service or protocol standard;

[-] indicates that the parameter shall be absent;

[*] indicates that the value is to be inserted by the tester before test execution.

For PDU's received by the tester:

[?] indicates that the tester need make no verification of the value of the parameter;

[-] indicates that the tester shall check that the parameter is absent.

Note – That the "?" and "-" symbols in value assignments of PDU components have got other meanings than "?" and "-" in generic ASP schedules.

A.5 *Test case identification*

Test cases are completely identified using four components:

- a test group identifier;
- a subgroup identifier;
- a validity identifier;
- a test number.

These four components are specified in two equivalent ways:

- as a Test Reference where the four components are textual and descriptive;

  Example: OriginalEncodedInfoTypeIndication/Recipient/Valid/2

- as a Test Identifier where the four components are numeric and concise.

  Example: 307.2.1.2.

A.5.1 *IPMS(P2) and MTS(P1) identification*

A.5.1.1 *Test Groups*

Number ranges have been allocated for the test groups as shown below:

Initial Tests	001 - 099
X.409 Tests	100 - 199
Protocol Elements Tests (for frequently occurring Elements)	200 - 299
X.400 Service Elements Tests	300 - 399
Additional Tests	400 - 499

A.5.1.2 *Subgroups*

Numeric identifiers have been allocated to the test subgroups as shown below:

Originator	1
Recipient	2
Encode	1
Decode	2
Relay	3
Relaying-Recipient	4
Relaying-Originator	5

A.5.1.3 *Validity identifiers*

Test cases which exercise valid behaviour are distinguished from those which exercise the IUT's reaction to invalid behaviour using the numeric identifiers shown below:

Valid	1
Invalid	2

A.5.1.4 *Test case numbers*

Test cases for a particular group/subgroup/validity are numbered sequentially.

A.5.2 *RTS identification*

A.5.2.1 *Test groups*

Number ranges have been allocated for the test groups as shown below:

Association Establishment	1
Association Release	2
Data Transfer	3
Association Recovery	4
X.409 Tests	5

A.5.2.2 *Subgroups*

Numeric identifiers have been allocated to the RTS subgroups as shown below:

Initiator	1
Responder	2
Sender	1
Receiver	2

A.5.2.3 *Validity identifiers*

Test cases which exercise valid behaviour are distinguished from those which exercise the IUT's reaction to invalid and inopportune behaviour using the numeric identifiers shown below:

Valid	1
Invalid	2
Inopportune	3

A.6 *Cross referencing*

A.6.1 *Cross reference numbering*

The MTS (P1) and IPMS (P2) test suites contain a cross referencing system for the ASPs, test PDUs and library components. The cross referencing appears in the left and right margins of the test suite as shown in Figure A-14/X.403.

Numbers in the left hand margin of the test suite are in sequential order and are "place identifiers". They occur whenever an ASP, test PDU or library component occurs in the test suite.

Whenever an ASP, test PDU or library component occurs, numbers are also placed in the right hand margin. These numbers are forward and backward references to the place identifiers of the other occurrences of the ASP, test PDU or library component.

Where a forward or backward reference can not be found then a dot (".") is printed in the right hand margin. This should not occur in fully defined test suites.

Where a line in the test suite contains more than one ASP, test PDU or library component, the cross references for each item in the line are separated by vertical bars ("|") in the right hand margin as shown in Figure A-15/X.403.

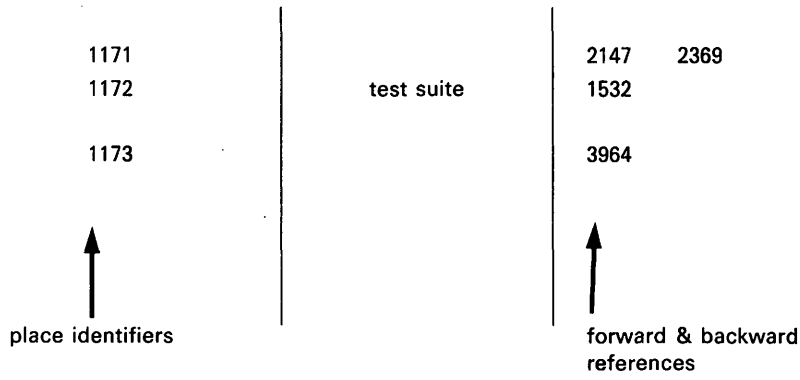


FIGURE A-14/X.403

Cross referencing

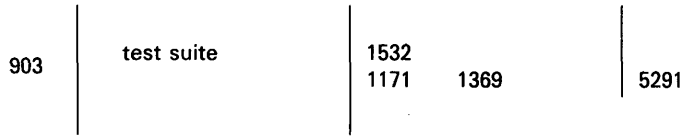


FIGURE A-15/X.403

Multiple cross references

A.6.2 *Cross reference listing*

At the end of MTS (P1) and IPMS (P2) test suites there is a separate cross reference listing of all the ASPs, test PDUs and library components together with the place identifiers of all their occurrences in the test suite.

Example:

```
:
:
:
IM_UAPDU_1_14          586 1467
IM_UAPDU_1_15          587 1470
:
:
:
```

The numbers on the right side indicate the places where the item occurs in the test suite.

ANNEX B

(to Recommendation X.403)

IPMS (P2) PICS proformas

B.1 *General*

As a prerequisite to conformance testing, the supplier of an IPMS (P2) implementation must provide a Protocol Implementation Conformance Statement (PICS).

The proforma IPMS (P2) PICS contained in this Annex specifies the information to be supplied.

This information is needed for test case selection. Suppliers should note that tests will be performed to check that services shown as not supported are in fact not present rather than improperly implemented.

The IPMS (P2) PICS is in two parts:

- a part requesting information concerning the support of service elements;
- a part requesting information concerning the support of protocol elements.

Information on service element support is requested in tabular form where, for each service element:

- the status of the service element is indicated as mandatory (M), optional (O) or not applicable (–) in columns labelled "STD";
- the actual support of the service element by the implementation on origination and reception is indicated by the supplier in columns labelled "IMP".

Information on protocol element support is requested in tabular form where, for each protocol element:

- the status of the protocol element on origination and reception is indicated as mandatory (M) or optional (O) in columns labelled "STD";
- any implementation constraints are indicated in the column labelled "CONST STD" where constraints are interpreted as a minimum for reception and a maximum for origination;
- the actual support of the protocol element by the implementation on origination and on reception is indicated by the supplier in the column labelled "STATUS IMP";
- the actual constraints of the implementation on origination and on reception is indicated by the supplier in the columns labelled "CONST IMP".

  Constraints may be expressed as a length or size (octets, bits, ...), a value (32k – 1) or a number of occurrences (4) depending on the element being constrained.

B.2 *IPMS (P2) PICS service elements proforma*

The requirements of the X.400 Recommendations are shown in the STD columns of the proforma B.1/X.403 using the following keys:

M Mandatory element (X.401 Basic or Essential Options)

O Optional element (X.401 Additional Options)

– Not applicable service element.

Suppliers of an implementation should use the IMP columns in the proforma to specify information concerning the support of service elements. For convenience, it is suggested that suppliers need only indicate with an "X" those service elements that are not supported.

B.3 *IPMS(P2) PICS protocol elements proformas*

The requirements of the X.400 Recommendations are shown in the STATUS STD columns of the proformas in Tables B-2/X.403 to B-4/X.403 using the following keys:

M Mandatory element (X.401 Basic or Essential Optional)

O Optional element (X.401 Additional Optional)

TABLE B-1/X.403

IPMS (P2) service elements proforma

SERVICE ELEMENT	ORIGINATION		RECEPTION	
	STD	IMP	STD	IMP
IP Message Identification	M		M	
Typed body	M		M	
Blind copy recipient indication	O		M	
Non receipt notification	O		O	
Receipt notification	O		O	
Auto forwarded indication	O		M	
Originator indication	M		M	
Authorizing users indication	O		M	
Primary-copy recipients indication	M		M	
Expiry date indication	O		M	
Cross referencing indication	O		M	
Importance indication	O		M	
Obsoleting indication	O		M	
Sensitivity indication	O		M	
Subject indication	M		M	
Replying IP message indication	M		M	
Reply request indication	O		M	
Forwarded IP message indication	O		M	
Body part encryption indication	O		M	
Multipart body	O		M	
Alternate recipient allowed	O		O	
Conversion prohibition	M		M	
Deferred delivery	M		–	
Deferred delivery cancellation	O		–	
Delivery notification	M		–	
Disclosure of other recipients	O		M	
Explicit conversion	O		–	
Grade of delivery selection	M		M	
Multi-destination delivery	M		–	
Prevention of non-delivery notification	O		–	
Probe	O		–	
Return of contents	O		–	
Alternate recipient assignment	–		O	
Hold for delivery	–		O	
Implicit conversion	O		O	

Protocol elements which correspond directly to service elements are indicated as mandatory if their corresponding service elements are shown in X.401 (1984) as Basic or Essential Optional, and as optional if their corresponding service elements are shown in X.401 (1984) as Additional Optional. Other protocol elements are indicated as mandatory or optional according to their designation in the UAPDU definitions in X.420 (1984).

The pragmatic constraints of the X.400 Implementor's Guide are shown in the CONST STD columns of the proformas in Tables B-2/X.403 to B-4/X.403.

Suppliers of an implementation should use:

- the STATUS IMP columns in each proforma to specify information concerning the support of protocol elements. For convenience, it is suggested that suppliers need only indicate with an "X" those protocol elements that are not supported;
- the CONST IMP columns in each proforma to specify the actual constraints of the implementation.

TABLE B-2/X.403

ORDescriptor proforma

ORDescriptor	ORIGINATION			RECEPTION			CONST STD Octets
	STATUS		CONST	STATUS		CONST	
	STD	IMP	IMP	STD	IMP	IMP	
ORName and/or FreeFormName	M			M			
ORName	O			O			
StandardAttributeList	M			M			
CountryName	O			O			
PrintableString	M			M			2
NumericString	M			M			3
ADMDName	O			O			
PrintableString	M			M			16
NumericString	M			M			16
X121Address	O			O			15
TerminalID	O			O			24
PrivateDomainName	O			O			
PrintableString	M			M			16
NumericString	M			M			16
OrganizationName	O			O			64
UniqueUAIdentifier	O			O			32
PersonalName	O			O			64
surname	M			M			40
givenName	O			O			16
initials	O			O			5
generationQualifier	O			O			3
OrganizationalUnit	O			O			32
DomainDefinedAttrList	O			O			4
type	M			M			8
value	M			M			128
FreeFormName	O			O			128
TelephoneNumber	O			O			32

TABLE B-3/X.403

IM-UAPDU proforma

UAPDU NAME: IM-UAPDU	ORIGINATION			RECEPTION			CONST
	STATUS		CONST	STATUS		CONST	
	STD	IMP	IMP	STD	IMP	IMP	STD
HEADING	M			M			
IPMessageID	M			M			
ORName	O			O			
PrintableString	M			M			64
Originator (ORDescr.)	M			M			
AuthorizingUser (ORDescr.)	O			M			
PrimaryRecipient (ORDescr)	M			M			
ReportRequest	O			O			
ReplyRequest	O			O			
CopyRecipient (ORDescr.)	M			M			
ReportRequest	O			O			
ReplyRequest	O			O			
BlindCopyRecipient (ORDes)	O			M			
ReportRequest	O			O			
ReplyRequest	O			O			
InReplyTo (IPMessageId)	M			M			
ORName	O			O			
PrintableString	M			M			64
Obsoletes (IPMessageId)	O			M			
ORName	O			O			
PrintableString	M			M			64
CrossReference (IPM.Id)	O			M			
ORName	O			O			
PrintableString	M			M			64
Subject	M			M			256
ExpiryDate	O			M			
ReplyBy	O			M			
ReplyToUser (ORDescr.)	O			M			
Importance	O			M			
Sensitivity	O			M			
Autoforwarded	O			M			
Body	M			M			
IA5Text	O			O			
repertoire	O			O			
IA5String	M			M			

TABLE B-3/X.403 *(suite)*

UAPDU NAME: IM-UAPDU	ORIGINATION			RECEPTION			CONST
	STATUS		CONST	STATUS		CONST	
	STD	IMP	IMP	STD	IMP	IMP	STD
G3Fax	O			O			
NumberOfPages	O			O			
G3NonBasicParams	O			O			
BitString	M			M			
TTX	O			O			
NumberOfPages	O			O			
TelexCompatible	O			O			
TeletexNonBasicParams	O			O			
GraficCharacterSet	O			O			
PageFormats	O			O			
MiscTerminalCapability	O			O			
PrivateUse	O			O			
T61String	M			M			
TIF.O	O			O			
T73Document	M			M			
TIF.1	O			O			
T73Document	M			M			
Videotex	O			O			
Videotexstring	M			M			
SFD	O			O			
SFD.Document	M			M			
TLX	O			O			
Voice	O			O			
Bit string	M			M			
Encrypted	O			O			
Bit string	M			M			
NationallyDefined	O			O			
ForwardedIpMessage	O			O			
Delivery	O			O			
DeliveryEnvelope	O			O			
ContentType	M			M			
Originator	M			M			
Original	M			M			
Priority	O			O			
DeliveryFlags	M			M			
OtherRecipients	O			O			
ThisRecipient	M			M			
IntendedRecipient	O			O			
Concreted	O			O			
Submission	M			M			
IM-UAPDU	M			M			

TABLE B-4/X.403

SR-UAPDU proforma

UAPDU NAME: SR-UAPDU	ORIGINATION			RECEPTION			CONST
	STATUS		CONST	STATUS		CONST	
	STD	IMP	IMP	STD	IMP	IMP	STD
NonReceiptInformation	M			M			
Reason	M			M			
NonReceiptQualifier	O			O			
Comments	O			O			
Returned	O			O			256
ReceiptInformation	M			M			
ReceiptTime	M			M			
TypeOfReceipt	O			O			
SupplementaryInformation	O			O			64
Reported (IPMessageId)	M			M			
ORName	O			O			
PrintableString	M			M			64
ActualRecipient (ORDescr.)	O			O			
IntendedRecipient (ORDescr)	O			O			
Converted	O			O			

ANNEX C

(to Recommendation X.403)

MTS (P1) PICS proformas

C.1 *General*

As a prerequisite to conformance testing, the supplier of an MTS (P1) implementation must provide a Protocol Implementation Conformance Statement (PICS).

The proforma MTS (P1) PICS contained in this Annex specifies the information to be supplied.

This information is needed for test case selection. Suppliers should note that tests will be performed to check that services shown as not supported are in fact not present rather than improperly implemented.

The MTS (P1) PICS is in two parts:

- a part requesting information concerning the support of service elements;
- a part requesting information concerning the support of protocol elements.

Information on service element support is requested in tabular form where, for each service element:

- the status of the service element is indicated as mandatory (M), optional (O) or not applicable (–) in columns labelled "STD";
- the actual support of the service element by the implementation on origination and reception is indicated by the supplier in columns labelled "IMP".

Information on protocol element support is requested in tabular form where, for each protocol element:

- the status of the protocol element on origination and reception is indicated as mandatory (M) or optional (O) in columns labelled "STD";
- any implementation constraints are indicated in the column labelled "CONST STD" where constraints are interpreted as a minimum for reception and a maximum for origination;
- the actual support of the protocol element by the implementation on origination and on reception is indicated by the supplier in the column labelled "STATUS IMP";
- the actual constraints of the implementation on origination and on reception is indicated by the supplier in the columns labelled "CONST IMP".

  Constraints may be expressed as a length or size (octets, bits, ...), a value (32k – 1) or a number of occurrences (4) depending on the element being constrained.

C.2 *Originator/recipient/relay capability*

Suppliers of an implementation should specify Originator/Recipient/Relay capabilities in the IMPLEMENTED column of Table C-1/X.403.

TABLE C-1/X.403

CAPABILITY	IMPLEMENTED
Originator	
Recipient	
Relay	

C.3 *MTS (P1) PICS service elements proforma*

The requirements of the X.400 Recommendations are shown in the STD columns of the proforma using the following keys:

M Mandatory element (X.401 Basic or Essential Optional)

O Optional element (X.401 Additional Optional)

– Not applicable service element.

Suppliers of an implementation should use the IMP columns in the proforma to specify information concerning the support of service elements. For convenience, it is suggested that suppliers need only indicate with an "X" those service elements that are not supported.

TABLE C-2/X.403

MTS (P1) Service elements proforma

SERVICE ELEMENT	ORIGINATION		RECEPTION		RELAY	
	STD	IMP	STD	IMP	STD	IMP
Content type indication	M		M		–	
Convert indication	M		M		M	
Delivery time stamp indication	M		M		–	
Message identification	M		M		–	
Non-delivery notification	M		M		M	
Original encoded information types indication	M		M		–	
Registered encoded information types	M		M		–	
Submission time stamp indication	M		M		–	
Alternate recipient allowed	M		O		–	
Deferred delivery	M		–		–	
Deferred delivery cancellation	M		–		–	
Delivery notification	M		M		–	
Disclosure of other recipients	M		M		M	
Grade of delivery selection	M		M		–	
Multidestination delivery	M		M		M	
Prevention of non-delivery notification	O		O		O	
Return of contents	O		O		O	
Conversion prohibition	M		M		–	
Explicit conversion	O		O		O	
Implicit conversion	O		O		O	
Probe	M		M		M	
Hold for delivery	–		O		–	
Alternate recipient assignment	–		O		–	

C.4 *MTS (P1) protocol elements proformas*

The requirements of the X.400 Recommendations are shown in the STATUS STD column of the proformas in Tables C-3/X.403 to C-6/X.403 using the following keys:

M Mandatory element (X.401 Basic or Essential Optional)

O Optional element (X.401 Additional Optional)

In the tables below, protocol elements which correspond directly to service elements are indicated as mandatory if their corresponding service elements are shown in X.401 (1984) as Basic or Essential Optional, and as optional if their corresponding service elements are shown in X.401 (1984) as Additional Optional. Other protocol elements are indicated as mandatory or optional according to their designation in the MPDU definitions in X.411 (1984).

For relay functions, protocol elements are indicated as mandatory or optional based only on their status in the P1 protocol specification.

The pragmatic constraints of the X.400 Implementor's Guide are shown in the CONS STD columns of the proformas in Tables C-3/X.403 to C-6/X.403.

Suppliers of an implementation should use:

- the STATUS IMP column in each proforma to specify information concerning the support of protocol elements. For convenience, it is suggested that suppliers need only indicate with an "X" those protocol elements that are not supported;
- the CONS IMP columns in each proforma to specify the actual constraints of the implementation.

TABLE C-3/X.403

ORName and EncodedInformationType proforma

	ORIGINATION			RECEPTION			RELAY			CONST
	STATUS		CONST	STATUS		CONST	STATUS		CONST	
	STD	IMP	IMP	STD	IMP	IMP	STD	IMP	IMP	STD
ORName	M			M			M			
StandardAttributeList	M			M			M			
CountryName	O			O			O			
NumericString	M			M			M			3 Ch
PrintableString	M			M			M			3 Ch
AdministrationDomainName	O			O			O			
NumericString	M			M			M			16 Ch
PrintableString	M			M			M			16 Ch
X121Address	O			O			O			15 Ch
TerminalID	O			O			O			24 Ch
PrivateDomainName	O			O			O			
NumericString	M			M			M			16 Ch
PrintableString	M			M			M			16 Ch
OrganizationName	O			O			O			64 Ch
UniqueUAIdentifier	O			O			O			32 Ch
PersonalName	O			O			O			64 Ch
surname	M			M			M			40 Ch
givenName	O			O			O			16 Ch
initials	O			O			O			5 Ch
generationQualifier	O			O			O			3 Ch
OrganizationalUnit	O			O			O			32 Ch
DomainDefinedAttributeList	O			O			O			8 Ch
type	M			M			M			
value	M			M			M			128 Ch
EncodedInformationType										
BITSTRING	M			M			M			32 bit
G3NonBasisParams	O			O			O			
TeletexNonBasisParams	O			O			O			
PresentationCapabilities	O			O			O			

TABLE C-4/X.403

UserMPDU proforma

MPDU NAME: UserMPDU	ORIGINATION			RECEPTION			RELAY			CONST
	STATUS		CONST	STATUS		CONST	STATUS		CONST	
	STD	IMP	IMP	STD	IMP	IMP	STD	IMP	IMP	STD
UMPDU ENVELOPE	M			M			M			
MPDUIdentifier	M			M			M			
GlobalDomainIdentifier	M			M			M			
CountryName	M			M			M			
NumericString	M			M			M			3 Ch
PrintableString	M			M			M			3 Ch
AdministrationDomainName	M			M			M			
NumericString	M			M			M			16 Ch
PrintableString	M			M			M			16 Ch
PrivateDomainName	O			O			O			
NumericString	M			M			M			16 Ch
PrintableString	M			M			M			16 Ch
IA5String	M			M			M			32 Ch
Originator	M			M			M			
OriginalEncodedInformationType	M			M			M			
ContentType	M			M			M			16 Ch
UaContentID	O			O			O			
Priority	M			M			M			
PerMessageFlag	M			M			M			16 b
DisclosureRecipients	M			M			M			
ConversionProhibited	M			M			M			
AlternateRecipientAllowed	M			O			O			
ContentReturnRequest	O			O			O			
DeferredDelivery	M			O			O			
PerDomainBilateralInfo	O			O			O			
CountryName	M			M			M			
NumericString	M			M			M			3 Ch
PrintableString	M			M			M			3 Ch
AdministrationDomainName	M			M			M			
NumericString	M			M			M			16 Ch
PrintableString	M			M			M			16 Ch
BilateralInfo	M			M			M			a)

TABLE C-4/X.403 *(suite)*

MPDU NAME: UserMPDU	ORIGINATION			RECEPTION			RELAY			CONST
	STATUS		CONST	STATUS		CONST	STATUS		CONST	
	STD	IMP	IMP	STD	IMP	IMP	STD	IMP	IMP	STD
RecipientInfo	M			M			M			
Recipient	M			M			M			
ExtensionIdentifier	M			M			M			b)
PerRecipientFlag	M			M			M			
ResponsabilityFlag	M			M			M			
ReportRequest	M			M			M			
UserReportRequest	M			M			M			
ExplicitConversion	O			O			O			
TraceInformation	M			M			M			
GlobalDomainIdentifier	M			M			M			
DomainSuppliedInfo	M			M			M			
Arrival	M			M			M			
Deferred	M			M			M			
Action relayed or rerouted	M			O			O			
Converted	O			O			O			
Previous	O			O			O			
UMPDU-CONTENT	M			M			M			

a) 1024 octets.

b) Max value 32k – 1.

TABLE C-5/X.403

DeliveryReportMPDU proforma

MPDU NAME: DeliveryReportMPDU	ORIGINATION			RECEPTION			RELAY			CONST
	STATUS		CONST	STATUS		CONST	STATUS		CONST	
	STD	IMP	IMP	STD	IMP	IMP	STD	IMP	IMP	STD
DELIVERYREPORTENVELOPE	M			M			M			
Report	M			M			M			
GlobalDomainIdentifier	M			M			M			
CountryName	M			M			M			
NumericString	M			M			M			3 Ch
PrintableString	M			M			M			3 Ch
AdministrationDomainName	M			M			M			
NumericString	M			M			M			16 Ch
PrintableString	M			M			M			16 Ch
PrivateDomainName	O			O			O			
NumericString	M			M			M			16 Ch
PrintableString	M			M			M			16 Ch
IA5String	M			M			M			32 Ch
Originator	M			M			M			
TraceInformation	M			M			M			
GlobalDomainIdentifier	M			M			M			
DomainSuppliedInfo	M			M			M			
Arrival	M			M			M			
Deferred	M			M			M			
Action relayed or rerouted	M			O			O			
Converted	O			O			O			
Previous	O			O			O			
DELIVERYREPORTCONTENT	M			M			M			
Original MPDUIdentifier	M			M			M			
GlobalDomainIdentifier	M			M			M			
CountryName	M			M			M			
NumericString	M			M			M			3 Ch
PrintableString	M			M			M			3 Ch
AdministrationDomainName	M			M			M			
NumericString	M			M			M			16 Ch
PrintableString	M			M			M			16 Ch
PrivateDomainName	O			O			O			
NumericString	M			M			M			16 Ch
PrintableString	M			M			M			16 Ch
IA5String	M			M			M			32 Ch
Intermediate	O			O			O			
TraceInformation	M			M			M			
GlobalDomainIdentifier	M			M			M			
DomainSuppliedInfo	M			M			M			
Arrival	M			M			M			
Deferred	M			M			M			
Action relayed or rerouted	M			O			O			
Converted	O			O			O			
Previous	O			O			O			

TABLE C-5/X.403 *(suite)*

MPDU NAME: DeliveryReportMPDU	ORIGINATION			RECEPTION			RELAY			CONST
	STATUS		CONST	STATUS		CONST	STATUS		CONST	
	STD	IMP	IMP	STD	IMP	IMP	STD	IMP	IMP	STD
UAContentId	O			O			O			
ReportRecipientInfo	M			M			M			
Recipient	M			M			M			
ExtensionIdentifier	M			M			M			32k – 1
PerRecipientFlag	M			M			M			
ResponsabilityFlag	M			M			M			
ReportRequest	M			M			M			
UserReportRequest	M			M			M			
LastTraceInformation	M			M			M			
arrival	M			M			M			
converted	O			O			O			
Report	M			M			M			
DeliveredInfo	M			M			M			
Delivery	M			M			M			
TypeOfUA	O			O			O			
NonDeliveredInfo	M			M			M			
ReasonCode	M			M			M			
DiagnosticCode	O			O			O			
IntendedRecipient	O			O			O			
SupplementaryInform.	O			O			O			64 Ch
Returned	O			O			O			
BillingInformation	O			O			O			[a]

[a] 1024 octets.

TABLE C-6/X.403

ProbeMPDU proforma

MPDU NAME: ProbeMPDU	ORIGINATION			RECEPTION			RELAY			CONST
	STATUS		CONST	STATUS		CONST	STATUS		CONST	
	STD	IMP	IMP	STD	IMP	IMP	STD	IMP	IMP	STD
PROBE ENVELOPE	M			M			M			
Probe	M			M			M			
GlobalDomainIdentifier	M			M			M			
CountryName	M			M			M			
NumericString	M			M			M			3 Ch
PrintableString	M			M			M			3 Ch
AdministrationDomainName	M			M			M			
NumericString	M			M			M			16 Ch
PrintableString	M			M			M			16 Ch
PrivateDomainName	O			O			O			
NumericString	M			M			M			16 Ch
PrintableString	M			M			M			16 Ch
IA5String	M			M			M			32 Ch
Originator	M			M			M			
ContentType	M			M			M			
UAContentId	O			O			O			
OriginalEncodedInformationType	M			M			M			
TraceInformation	M			M			M			
GlobalDomainIdentifier	M			M			M			
DomainSuppliedInfo	M			M			M			
Arrival	M			M			M			
Deferred	M			M			M			
Action relayed or rerouted	M			O			O			
Converted	O			O			O			
Previous	O			O			O			
PerMessageFlag	M			M			M			
DisclosureRecipients	M			M			M			
ConversionProhibited	M			M			M			
AlternateRecipientAllowed	M			O			O			
ContentReturnRequest	O			O			O			
ContentLength	O			O			O			
PerDomainBilateralInfo	O			O			O			
CountryName	M			M			M			
NumericString	M			M			M			3 Ch
PrintableString	M			M			M			3 Ch
AdministrationDomainName	M			M			M			
NumericString	M			M			M			16 Ch
PrintableString	M			M			M			16 Ch
BilateralInfo	M			M			M			a)
RecipientInfo	M			M			M			
Recipient	M			M			M			
ExtensionIdentifier	M			M			M			b)
PerRecipientFlag	M			M			M			
ResponsibilityFlag	M			M			M			
ReportRequest	M			M			M			
UserReportRequest	M			M			M			
ExplicitConversion	O			O			O			

a) 1024 octets.

b) Max value 32k – 1.

ANNEX D

(to Recommendation X.403)

RTS PICS proformas

D.1 *General*

As a prerequisite to conformance testing of an RTS implementation, the supplier must provide a Protocol Implementation Conformance Statement (PICS).

The proforma RTS PICS contained in this Annex specifies the information to be supplied.

This information is needed for test case selection. Suppliers should note that tests will be performed to check that services shown as not supported are in fact not present rather than improperly implemented.

The RTS PICS is in three parts:

- Two parts requesting information concerning the support of RTS service primitives.

  If primitives have only mandatory parameters, they should be marked as "not supported" if any of their parameters are not supported.
- A part requesting information concerning the support of protocol elements.

Information on service element support is requested in tabular form where, for each service element:

- the status of the service element is indicated as mandatory (M), optional (O), conditional (C) or not applicable (–) in columns labelled "STD";
- the actual support of the service element by the implementation as initiator or responder is indicated by the supplier in columns labelled "IMP".

Information on protocol element support is requested in tabular form where, for each protocol element:

- the status of the protocol element where the IUT is initiator or responder is indicated as mandatory (M) or optional (O) in columns labelled "STD";
- any implementation constraints are indicated in the column labelled "CONST STD" where constraints are interpreted as a minimum for reception and a maximum for origination;
- the actual support of the protocol element by the implementation as initiator or responder is indicated by the supplier in the column labelled "STATUS IMP";
- the actual constraints of the implementation as initiator or responder are indicated by the supplier in the columns labelled "CONST IMP".

  Constraints may be expressed as a length or size (octets, bits, ...) or a value (32) depending on the element being constrained.

D.2 *RTS PICS service primitives proforma*

The requirements of the X.400 Recommendations are shown in the STD columns of the proforma using the following keys:

M Mandatory element

O Optional element

Suppliers of an implementation should use the IMP columns in the proforma to specify information concerning the support of service elements. For convenience, it is suggested that suppliers need only indicate with an "X" those service primitives that are not supported.

D.3 *RTS PICS service parameters proforma*

RTS service parameters are mapped to Session and Presentation as below:

- The parameters of the OPEN.Request and the OPEN.Indication are mapped to the SCONNECT.Request and SCONNECT.Indication and to the corresponding PConnect.
- Responder/Initiator-address and Initial-turn are mapped to the SCONNECT.

– Dialogue-mode, Application-protocol and User-data are mapped to the PConnect.
– The parameters of the OPEN.Response and OPEN.Confirmation are all mapped to PAccept or PRefuse.

TABLE D-1/X.403

RTS service	INITIATOR		RESPONDER	
	STATUS		STATUS	
	STD	IMP	STD	IMP
OPEN	M		M	
CLOSE	M		M	
TURN-GIVE	O		O	
TURN-PLEASE	O		O	
TRANSFER	M		M	
EXCEPTION	M		M	

Since all OPEN service parameters are mapped to the PConnect protocol element (apart from Response-address and Initial-turn which are mandatory), there is an apparent duplication of information requested in Tables D-2/X.403 to D-5/X.403 with that requested in Table D-6/X.403.

Tables D-2/X.403 to D-5/X.403 are useful nevertheless because they make sure that all the mandatory parameters are really supported and because they make a static conformance review easier.

The requirements of the X.400 Recommendations are shown in the STD columns of the proforma using the following keys:

M Mandatory parameters

O Optional parameters

C Conditional parameters

– Not applicable service parameters

Suppliers of an implementation should use the IMP columns in the proforma to specify information concerning the support of service elements. For convenience, it is suggested that suppliers need only indicate with an "X" those service elements that are not supported.

D.4 *RTS protocol elements*

The requirements of the X.400 Recommendations are shown in the STATUS STD column of the proforma in Table D-6/X.403 to D-9/X.403 using the following keys:

M Mandatory element

O Optional element

The pragmatic constraints of the X.400 Implementor's Guide are shown in the CONST STD columns of the proforma in Table D-6/X.403 to D-9/X.403.

Suppliers of an implementation should use:

– the STATUS IMP column in the proforma to specify information concerning the support of protocol elements. For convenience, it is suggested that suppliers need only indicate with an "X" those protocol elements that are not supported;
– the CONST IMP columns in the proforma to specify the actual constraints of the implementation.

TABLE D-2/X.403

OPEN.Request	INITIATOR	
	STATUS	
	STD	IMP
OPEN.Request		
Responder-address	M	
Dialogue-mode	M	
monologue	M	
twa	O	
Initial-turn	M	
initiator	M	
responder	–	
Application-protocol	M	
P1	M	
User-data	C	

TABLE D-3/X.403

OPEN.Indication	RESPONDER	
	STATUS	
	STD	IMP
OPEN.Indication		
Initiator-address	M	
Dialogue-mode	M	
monologue	M	
twa	M	
Initial-turn	M	
initiator	M	
responder	–	
Application-protocol	M	
P1	M	
User-data	C	

TABLE D-4/X.403

OPEN.Response	INITIATOR	
	STATUS	
	STD	IMP
OPEN.Response		
Disposition	M	
accepted	C	
refused	C	
User-data	C	
Refusal-reason	C	
unacceptable dialogue mode	C	
authentication failure	C	
busy	C	

TABLE D-5/X.403

OPEN.Confirmation	RESPONDER	
	STATUS	
	STD	IMP
OPEN.Confirmation		
Disposition	M	
accepted	C	
refused	C	
User-data	C	
Refusal-reason	C	
unacceptable dialogue mode	C	
authentication failure	C	
busy	C	

For some parameters, only one value is applicable (e.g. DataTransferSyntax: O). There are other parameters (e.g. checkpointSize, RefuseReason) that may vary under various circumstances and run-time conditions. This information is available in a PIXIT and in such cases a reference to the PIXIT normally can be made in the constraints-field, if the parameter is not fixed.

In a recovery, the SessionConnectedId is used in the PConnect and the PAccept. This SessionConnectedID may or may not be encoded according to X.409. This information is not important for the PICS because it is not a criterion for the Static Conformance Review or for the Test Case Selection and would normally be given in a PIXIT.

TABLE D-6/X.403

PConnect	INITIATOR			RESPONDER			CONST
	STATUS		CONST	STATUS		CONST	
	STD	IMP	IMP	STD	IMP	IMP	STD
PConnect	M			M			
DataTransferSyntax	M			M			
pUserData	M			M			
checkpointSize	O			M			
windowSize	O			M			
dialogueMode	O			M			
monologue	M			M			
twa	O			M			
ConnectionData	M			M			512
open	M			M			
null	M			M			
MTAName	M			M			32 oct
Password	M			M			64 oct
recover	O			M			
SessionConnectionIden.	M			M			
CallingSSUserReferen.	M			M			64 oct
CommonReference	M			M			
AdditionalRef.Info.	O			O			4 oct
applicationProtocol	O			M			

TABLE D-7/X.403

PAccept	INITIATOR			RESPONDER			CONST
	STATUS		CONST	STATUS		CONST	
	STD	IMP	IMP	STD	IMP	IMP	STD
PAccept	M			M			
DataTransferSyntax	M			M			
PuserData	M			M			
checkpointSize	O			O			
windowSize	O			O			
ConnectionData	M			M			512
open	M			M			
null	M			M			
MTAName	M			M			32 oct
Password	M			M			64 oct
recover	O			O			
SessionConnectionIden.	M			M			
CallingSSUserReferen.	M			M			64 oct
CommonReference	M			M			
AdditionalRef.Info.	O			O			4 oct

TABLE D-8/X.403

PRefuse	INITIATOR			RESPONDER			CONST
	STATUS		CONST	STATUS		CONST	
	STD	IMP	IMP	STD	IMP	IMP	STD
PRefuse	M			M			
RefuseReason	M			M			
rtsBusy	C			M			
cannotRecover	C			M			
validationFailure	C			M			
unacceptableDialogueMode	C			M			

TABLE D-9/X.403

AbortInformation	INITIATOR			RESPONDER			CONST
	STATUS		CONST	STATUS		CONST	
	STD	IMP	IMP	STD	IMP	IMP	STD
AbortInformation	M			M			
AbortReason	O			O			
localSystemProblem	C			M			
invalidParameter	C			M			
unrecognizedActivity	C			M			
temporaryProblem	C			M			
protocolError	C			M			
transferCompleted	C			M			
reflectedParameter	O			O			

Recommendation X.407

MESSAGE HANDLING SYSTEMS: ABSTRACT SERVICE DEFINITION CONVENTIONS[1)]

(Melbourne, 1988)

The establishment in various countries of telematic services and computer-based store-and-forward message services in association with public data networks creates a need to produce standards to facilitate international message exchange between subscribers to such services.

The CCITT,

considering

(a) the need for Message Handling Systems;

(b) that Message Handling is a complex distributed information processing task;

(c) that a means for abstractly defining such tasks is required;

(d) that Recommendation X.200 defines the Reference Model of Open Systems Interconnection for CCITT applications;

(e) that Recommendations X.208, X.217, X.218, and X.219 provide the foundation for CCITT applications,

unanimously declares

(1) that conventions for defining abstract services are defined in Section 2;

(2) that techniques for the realization of abstract services so defined are discussed in Section 3.

TABLE OF CONTENTS

[1)] Recommendation X.407 and ISO 10021-3, Information processing systems – Text Communication – MOTIS – Abstract Service Definition Conventions, were developed in close collaboration and are technically aligned.

SECTION 1 – INTRODUCTION

0 Introduction

This Recommendation is one of a set of Recommendations for Message Handling. The entire set provides a comprehensive blueprint for a Message Handling System (MHS) realized by any number of cooperating open systems.

The Message Handling System made possible by these Recommendations is a complex distributed information processing task, many of whose components themselves have these characteristics.

This Recommendation specifies the conventions for defining the distributed information processing tasks of Message Handling and may also be useful for other applications.

The text of this Recommendation is the subject of joint CCITT-ISO agreement. The corresponding ISO specification is ISO 10021-3.

1 Scope

This Recommendation specifies the conventions used to specify the distributed information processing tasks that arise in Message Handling.

This Recommendation is structured as follows. Section 1 is this introduction. Section 2 specifies the conventions for defining a distributed information processing task abstractly. Section 3 gives principles for realizing the communication aspects of such tasks concretely, such as by Open Systems Interconnection (OSI) protocols. Annexes provide important supplemental information.

There are no requirements for conformance to this Recommendation.

2 References

This Recommendation cites the documents below.

Recommendation X.200 Reference model of open systems interconnection for CCITT applications (see also ISO 7498).

Recommendation X.208 Specification of abstract syntax notation one (ASN.1) (see also ISO 8824).

Recommendation X.209 Specification of basic encoding rules for abstract syntax notation one (ASN.1) (see also ISO 8825).

Recommendation X.217 Association control service definition for open systems interconnection for CCITT applications (see also ISO 8649).

Recommendation X.219 Remote operations: Model, notation and service definition (see also ISO 9072-1).

3 Definitions

For the purposes of this Recommendation, the definition of Annex E and below apply.

This Recommendation is based upon the concepts developed in Recommendation X.200 and uses the following terms defined in it:

a) abstract syntax;
b) Application Layer;
c) application protocol data unit (APDU);
d) application protocol;
e) application service element (ASE);
f) concrete transfer syntax;
g) distributed information processing task;
h) layer service;
i) layer;
j) open system;
k) Open Systems Interconnection (OSI);
l) real open system.

This Recommendation uses the following terms defined in Recommendation X.208:

a) Abstract Syntax Notation One (ASN.1);
b) (data) type;
c) (data) value;
d) import;
e) Integer;
f) macro;
g) module;
h) Object Identifier;
i) tag.

This Recommendation uses the following terms defined in Recommendation X.209:

a) Basic Encoding Rules.

This Recommendation uses the following terms defined in Recommendation X.217:

a) application context (AC).

This Recommendation uses the following terms defined in Recommendation X.219:

a) bind operation;
b) error;
c) linked;
d) operation;
e) Remote Operation Service (ROS);
f) Remote Operations;
g) unbind operation.

4 Abbreviations

For the purposes of this Recommendation, the abbreviations of Annex E apply.

5 Conventions

This Recommendation uses the descriptive conventions identified below.

5.1 *ASN.1*

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

a) to define the OBJECT, PORT, and REFINE macros, the ASN.1 macro notation of Recommendation X.208;

b) to define the ABSTRACT-BIND, -UNBIND, -OPERATION, and -ERROR macros, the BIND, UNBIND, OPERATION, and ERROR macros of Recommendation X.219;

c) to specify the abstract syntax of information objects in the example of Annex A, ASN.1 itself;

d) to specify various *abstract models* in the example of Annex A, the OBJECT, PORT, and REFINE macros of § 7;

e) to specify various *abstract services* in the example of Annex A, the ABSTRACT-BIND, -OPERATION, and -ERROR macros of § 8.

ASN.1 appears both in the body of this Recommendation to aid the exposition and again, largely redundantly, in Annexes for reference. If differences are found between the two, a specification error is indicated.

Note – ASN.1 tags are implicit throughout the ASN.1 modules in the Annexes; the modules are definitive in that respect.

5.2 *Terms*

Throughout this Recommendation, terms are rendered in **bold** when defined, in *italic* when reference is made to them prior to their definitions, without emphasis upon all other occasions.

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

SECTION 2 – ABSTRACT SERVICE DEFINITION CONVENTIONS

6 Overview

When faced with the job of describing and specifying a complex distributed information processing task, one is wise to begin by specifying the task in abstract, rather than concrete terms. This approach ensures that the task's functional requirements are stated independently of its concrete realization. Such separation is important, among other reasons, because each aspect of the task may admit of *several* concrete realizations. In a Message Transfer System comprising three message transfer agents, e.g., the first and second might interact using OSI communication, the second and third by proprietary means.

This section specifies the conventions for abstractly describing a distributed information processing task both macroscopically and microscopically. The former description is called an abstract model, the latter an *abstract service.*

Various formal tools for specifying *abstract models* and *services* are defined in this section. A comprehensive example of their use is given in Annex A. The reader may wish to refer to that Annex while reading the present section.

This section covers the following topics:

a) *Abstract models.*

b) *Abstract services.*

Note – The formal tools mentioned above are neither a formal description language nor a substitute for such. They are simply an ASN.1 notation that supports the informal descriptive conventions defined in this section.

7 Abstract models

A macroscopic description of a distributed information processing task is called an **abstract model (model)** of that task and of the environment in which it is carried out. It is based upon the concepts of *abstract objects*, *ports*, *services*, and *refinements*. (The concept of an *abstract service* is much more fully developed in § 8.)

7.1 *Abstract objects*

An **abstract object (object)** is a functional entity, one of perhaps several which interact with one another. Objects are of different types which determine their function and behavior. An object of one type, e.g., might represent a system, multiple objects of another type its users. Objects interact by means of *abstract ports*.

An object type is specified by means of the OBJECT macro. Such a specification lists the types of *abstract ports* that provide access to such an object. For each *asymmetric* port type, the specification indicates whether the ports of that type are *consumer* or *supplier* ports.

```
OBJECT MACRO ::=
BEGIN

TYPE NOTATION   ::= "PORTS" "{" PortList "}" | empty
VALUE NOTATION  ::= value (VALUE OBJECT IDENTIFIER)

PortList        ::= Port "," PortList | Port
Port            ::= value (PORT) PortType

PortType        ::= Symmetric | Asymmetric

Symmetric       ::= empty
Asymmetric      ::= Consumer | Supplier

Consumer        ::= "[C]"

Supplier        ::= "[S]"

END
```

A data value of type OBJECT is an Object Identifier that unambiguously and uniquely identifies the specified object type.

Note – The keyword "OBJECT" is reserved in ASN.1. Selection of a suitable replacement for use in the present context is for further study.

7.2 *Abstract ports*

An **abstract port (port)** is a point at which an abstract object interacts with another abstract object. Ports are of different types which determine the kinds of interactions they enable. Ports of one type, e.g., might represent the means by which a directory system is accessed, ports of another type the means by which it is administered.

Port types are themselves of the following two varieties:

a) **symmetric**: All instances of a symmetric port type are identical;

b) **asymmetric**: Each instance of an asymmetric port type is of one of two kinds, **supplier** and **consumer**.

Note – A particular allocation of the terms "supplier" and "consumer" is often intuitive. One might naturally consider a file system, e.g., to present supplier ports to its users and administrators. Strictly speaking, however, the assignment of the two terms is arbitrary.

Two objects can interact with one another by means of a port in one and a port in the other only while those ports are in contact with one another, or **bound**. The actions by means of which this state is initiated and terminated for one or more port pairs are called **binding** and **unbinding**, respectively.

Two ports can be bound only if they **match**. Any two ports of the same, symmetric type match. Two ports of the same, asymmetric type match if and only if one is a supplier, the other a consumer.

A port type is specified by means of the PORT macro. Such a specification identifies the *abstract operations* that represent the interactions possible while two such ports are bound. If none is listed, the *abstract operations* shall be considered unspecified.

```
PORT MACRO ::=
BEGIN

TYPE NOTATION    ::= Operations | empty
VALUE NOTATION   ::= value (VALUE OBJECT IDENTIFIER)

Operations       ::= Symmetrical | Asymmetrical

Symmetrical      ::= "ABSTRACT" "OPERATIONS" "{" OperationList "}"
Asymmetrical     ::= OneSided | TwoSided

OneSided         ::= Consumer | Supplier
TwoSided         ::= Consumer Supplier | Supplier Consumer

Consumer         ::= "CONSUMER" "INVOKES" "{" OperationList "}"
Supplier         ::= "SUPPLIER" "INVOKES" "{" OperationList "}"

OperationList    ::= Operation "," OperationList | Operation
Operation        ::= value (ABSTRACT-OPERATION) | -- identifying the abstract operation
                                                    by data value
                     type -- identifying the abstract operation by data type

END
```

If the port type is symmetric, both objects offer all listed *abstract operations.* If the port type is asymmetric, the macro distinguishes between the *abstract operations* an object with the consumer port offers and those an object with the supplier port offers.

A data value of type PORT is an Object Identifier that unambiguously and uniquely identifies the specified port type.

7.3 *Abstract services*

An **abstract service** is the set of capabilities that one object offers to another by means of one or more of its ports. The former object is called an **abstract service provider (provider)**, the latter an **abstract service user (user)**. Each port in question may be either symmetric or asymmetric and, if the latter, either consumer or supplier.

An abstract service may have any number of users and providers.

Whenever the abstract service ports of a provider are bound to the matching ports of a user, an **abstract association** (or **association**) is said to exist between the two objects.

An abstract service is specified as indicated in § 8.

Note – An abstract service serves much the same purpose within the Application Layer as does one of the layer services of lower OSI layers.

7.4 *Abstract refinements*

An object can be viewed in different ways at different times. On some occasions it is convenient to think of an object as atomic. This is the case, e.g., when describing how an object interacts with other objects external to it, i.e., when specifying its abstract service. On other occasions, it may be more convenient to think of an object as composite, i.e., constructed from other objects. This might be the case, e.g., when describing how an object is realized.

Like any objects, component objects have ports. Some are those visible on the "surface" of the constructed object. Others enable the component objects to interact, thus supporting the provision and use of lesser abstract services among the component objects, which cooperate to provide the overall abstract service of the constructed object.

The functional decomposition of an object into several lesser objects is called the **abstract refinement (refinement)** of that object.

The technique of refinement can be applied recursively. A component object can itself be refined to reveal *its* internal structure. This can continue until one reaches component objects best considered atomic.

A refinement is specified by means of the REFINE macro. It identifies the object whose internal structure is being revealed and the component objects used in its construction. Each component object is characterized as either unique or recurring. The macro also indicates which ports of component objects are bound to ports of other component objects, and which are visible at the surface of the composite object.

```
REFINE MACRO ::=
BEGIN

TYPE NOTATION    ::= Object "AS" ComponentList
VALUE NOTATION   ::= value (VALUE OBJECT IDENTIFIER)

ComponentList    ::= Component ComponentList | Component
Component        ::= ObjectSpec Ports

ObjectSpec       ::= Object | Object "RECURRING"

Ports            ::= PortSpecList | empty
PortSpecList     ::= PortSpec PortSpecList | PortSpec
PortSpec         ::= value (PORT) PortSide PortStatus

PortSide         ::= Consumer | Supplier | empty
Consumer         ::= "[C]"
Supplier         ::= "[S]"

PortStatus       ::= "VISIBLE" | "PAIRED" "WITH" ObjectList

ObjectList       ::= Object "," ObjectList | Object
Object           ::= value (OBJECT)

END
```

A data value of type REFINE is an Object Identifier.

Note – As with objects themselves, ports, in principle, can be viewed in different ways at different times. On some occasions, it is convenient to think of a port (pair) as atomic. However, one can imagine refining a port itself to examine how communication of this type can be provided. In this view, a port pair is itself viewed as being supported by a collection of objects. This would enhance the ability to specify communications capabilities. This "port refinement" concept is not pursued further in this version of this Recommendation.

8 Abstract services

A microscopic description of a distributed information processing task is a specification of the abstract service that defines how the task is initiated, controlled, and terminated. It is based upon the concepts of *abstract bind operations*, *unbind operations*, *operations*, and *errors*, as well as the enabling concept of *abstract procedures*.

Note – The macros defined below imply use of ASN.1 to specify arguments, results, and parameters. Any context-specific tags, e.g., assigned in the course of the specifications, although meaningless in that context, play an important role in a ROS realization of the abstract service.

8.1 *Abstract procedures*

An **abstract procedure (procedure)** is a task that one object carries out at another's request. The making of the request and the carrying out of the task are called the **invocation** and **performance** of the procedure. The objects that issue and act upon the request are called the **invoker** and **performer**, respectively.

A procedure may (but need not) require that an invoker, upon invocation, supply to the performer a single information object of a prescribed type, which is called the procedure's **argument**.

Every performance of every procedure has an outcome, success or failure. A procedure is considered to succeed if it is carried out in full, to fail if it is terminated prematurely.

A procedure may (but need not) require that the performer apprise the invoker of success. It may (but need not) further require that it supply, when reporting success, a single information object of a prescribed type, which is called the procedure's **result**.

A procedure may (but need not) require that the performer apprise the invoker of failure. It may (but need not) further require that it supply certain information when reporting failure.

Note – In subsequent § ASN.1 is prescribed as the means for specifying the abstract syntax of the arguments and results of procedures (as well as of the parameters of *abstract errors*). These uses of ASN.1 do not imply that these information objects are *necessarily* transported between open systems. In particular, the fact that the information objects, by virtue of their description in ASN.1 and of its Basic Encoding Rules, have concrete transfer syntaxes is immaterial in the present context. ASN.1 is simply a convenient tool for formally describing the information objects' abstract syntax.

8.2 *Abstract bind operations*

An **abstract bind operation** is a procedure whose successful performance binds one or more pairs of abstract ports. The object which invokes an abstract bind operation is said to be the **initiator**, that which performs it the **responder**.

An abstract bind operation is suitable for binding a particular set of ports of the initiator to a matching set of the responder. Where one or more ports in the set are asymmetric, the abstract bind operation may be suitable for binding to the consumer side only, the supplier side only, or to either.

An abstract bind operation is a fully general procedure except that, if information is conveyed to the invoker upon failure, it is constrained to a single information object, called **error information**.

An abstract bind operation is specified by means of the ABSTRACT-BIND macro whose definition is as follows:

```
ABSTRACT-BIND MACRO ::=
BEGIN

TYPE NOTATION   ::= Ports Bind
VALUE NOTATION  ::= value (VALUE BindType)

Ports           ::= "TO" "{" PortList "}" | empty
PortList        ::= Port "," PortList | Port
Port            ::= value (PORT) PortSide
PortSide        ::= Consumer | Supplier | empty
Consumer        ::= "[C]"
Supplier        ::= "[S]"

Bind            ::= type (BindType) -- must be a BIND type
                    | empty <BindType ::= BIND>

END
```

The "Ports" clause, introduced by the keyword "TO", lists the ports of a responder which this abstract bind operation will bind. If an asymmetric port is listed there, without being qualified by "[S]" or "[C]", this means that the abstract bind operation is suitable for use in binding such a port in either direction.

Note that the specification of the argument, result, and/or error information is accomplished by means of an (embedded) BIND macro of Remote Operations, defined in Recommendation X.219, and it is a value of such a type that the macro returns. If none is provided, the default "BIND" is returned.

Note – The relationship of ABSTRACT-BIND and BIND can help make trivial the ROS realization of an abstract service; see § 10.1.

An abstract service typically comprises an abstract bind operation for each type of port involved in its provision. When several port types are involved, their abstract bind operations may but need not be distinct.

8.3 *Abstract unbind operations*

An **abstract unbind operation** is a procedure whose performance, successful or not, unbinds two ports. It is invoked by the object which invoked the corresponding abstract bind (i.e., the initiator) and performed by the responder.

An abstract unbind operation is suitable for unbinding a particular set of ports of the initiator from a matching set of the responder. Where one or more ports in the set are asymmetric, the abstract unbind operation may be suitable for unbinding from the consumer side only, the supplier side only, or either.

An abstract unbind operation is a fully general procedure except that, if information is conveyed to the invoker upon failure, it is constrained to a single information object, called **error information.**

An abstract unbind operation is specified by means of the ABSTRACT-UNBIND macro whose definition is as follows:

```
ABSTRACT-UNBIND MACRO ::=
BEGIN

TYPE NOTATION   ::= Ports Unbind
VALUE NOTATION  ::= value (VALUE UnbindType)

Ports           ::= "FROM" "{" PortList "}"
PortList        ::= Port "," PortList | Port
Port            ::= value (PORT) PortSide
PortSide        ::= Consumer | Supplier | empty
Consumer        ::= "[C]"
Supplier        ::= "[S]"

Unbind          ::= type (UnbindType) |
                    -- must be an UNBIND type
                    empty <UnbindType ::= UNBIND>.

END
```

The "Ports" clause, introduced by the keyword "FROM", lists the ports of a responder from which this abstract unbind operation will unbind. If an asymmetric port is listed there, without being qualified by "[S]" or "[C]", this means that the abstract unbind operation is suitable for use in unbinding such a port in either direction (although the actual direction is determined by the direction in which the bind took place).

Note that the specification of the argument, result, and/or error information is accomplished by means of an (embedded) UNBIND macro of Remote Operations, defined in Recommendation X.219, and it is a value of such a type that the macro returns. If none is provided, the default "UNBIND" is returned.

Note – The relationship of ABSTRACT-UNBIND and UNBIND helps make trivial the ROS realization of an abstract service; see § 10.1.

An abstract service typically comprises an abstract unbind operation for each type of port involved in its provision. When several port types are involved, their abstract unbind operations may but need not be distinct.

8.4 *Abstract operations*

An **abstract operation** is a procedure that may be invoked in the context of two bound ports. Its failure has no effect upon the binding. If the ports are assymmetric, whether the invoker is the object having the consumer port, the object having the supplier port, or either is prescribed by the port. If the ports are symmetric, the invoker may be either object. Whether the ports are symmetric or asymmetric, the remaining object is the performer.

An abstract operation is a fully general procedure except for the information conveyed to the invoker upon failure. An abstract operation fails when it encounters an *abstract error,* and the information conveyed is constrained to that required to report that abstract error. Whether failure is reported and, if so, which abstract errors can be encountered are prescribed for each abstract operation.

An abstract operation is specified by means of the ABSTRACT-OPERATION macro. Its definition is identical to that of the OPERATION macro of Remote Operations, specified in Recommendation X.219.

```
ABSTRACT-OPERATION MACRO ::= OPERATION
```

An abstract service comprises zero or more abstract operations for each type of port involved in its provision. When several port types are involved, they may but need not have abstract operations in common.

Note – The equivalence of ABSTRACT-OPERATION and OPERATION helps make trivial the ROS realization of an abstract service; see § 10.1.

8.5 *Abstract errors*

An **abstract error** is an exceptional condition that may arise during the performance of an abstract operation, causing it to fail.

When an abstract error is reported, the performer conveys to the invoker the identity of the abstract error and possibly a single information object called its **parameter.** Whether a parameter is returned and, if so, its type are prescribed for each abstract error.

An abstract error is specified by means of the ABSTRACT-ERROR macro. Its definition is identical to that of the ERROR macro of Remote Operations, specified in Recommendation X.219.

ABSTRACT-ERROR; MACRO ::= ERROR

An abstract service comprises the zero or more abstract errors reported by its abstract operations.

Note – The equivalence of ABSTRACT-ERROR and ERROR helps make trivial the ROS realization of an abstract service; see § 10.1.

SECTION 3 – ABSTRACT SERVICE REALIZATIONS

9 Overview

Once a distributed information processing task has been described and specified in abstract terms, the manner in which each aspect of the task is to be concretely realized must be prescribed. As suggested previously, each aspect may admit of several concrete realizations.

This section specifies principles for concretely realizing abstract models and services. A **real** x is the computer process or system, or the real open system that concretely realizes an abstract object of type x.

This section covers the following topics:

a) OSI realizations.

b) Proprietary realizations.

Note – The aspects of an abstract model stressed here are abstract ports and their bindings. This is because abstract ports mark the boundary not only between abstract objects but also between the physical systems that concretely realize those abstract objects. Thus abstract ports and bindings are the parts of an abstract model that must be constructed or constructable with OSI tools if open systems interworking is to occur.

10 OSI realizations

A primary objective of CCITT Recommendations and ISO Standards is to specify how distributed information processing tasks are realized when carried out by several cooperating real open systems.

In the OSI environment, objects are realized by means of application processes, with, in general, a many-to-many mapping of objects to application processes. Communication among objects which are realized by application processes in different open systems is accomplished by OSI application protocols (consisting of application contexts). An application context thus realizes the binding, use, and unbinding of a number of port pairs.

The specification of an application context is in terms of the coordinated operation of a number of application-service-elements. Realization is therefore particularly straightforward to specify if an application-service-element is defined to correspond to each port whose communication is to be supported.

The realization of abstract ports and bindings by means of ASEs and ACs is discussed below. Both ROS and non-ROS realizations are considered.

10.1 *ROS realizations*

The concrete realization of ports and bindings is often trivial when accomplished by means of Remote Operations.

This is true because it is straightforward to define an abstract service which is such that there exists a ROS-based application protocol that is functionally identical to it. This is true in turn because the framework for the specification of abstract services is isomorphic to that for the specification of ROS-based application protocols. The correspondences behind the isomorphism are listed in Table 1/X.407.

TABLE 1/X.407

Correspondences of abstract services and ROS-based protocols

Aspect of Abstract service	Aspect of ROS-based protocol
Abstract bind operation	Bind operation
Abstract unbind operation	Unbind operation
Abstract operation	Operation
Abstract error	Error

The correspondences of the table arise from the fact that corresponding aspects are formally specified using closely-related, or equivalent macros, as summarized in Table 2/X.407.

TABLE 2/X.407

Equivalent abstract service and ROS macros

Abstract service macro	ROS macro
ABSTRACT-BIND	BIND
ABSTRACT-UNBIND	UNBIND
ABSTRACT-OPERATION	OPERATION
ABSTRACT-ERROR	ERROR

The definition of ROS-based ASEs and ACs that concretely realize abstract ports is explored in Annex A by means of an example.

For the realization to be trivial, it is necessary that there be an abstract bind operation which binds all of the ports which must be paired.

Note – Where there is more than one port (pair) involved in the abstract service, this requires that the abstract bind operation be designed for the particular ports involved. There is (currently) no provision for the automatic synthesis of a suitable abstract bind based, e.g., upon the definitions of abstract bind operations defined for the individual ports.

10.2 *Non-ROS realizations*

The concrete realization of ports and bindings is a more substantial task when attempted by means other than Remote Operations, and little can be said about the general propositions.

Despite the above, the following two observations are relevant:

a) The concrete realization of an abstract service as an application protocol is greatly simplified by using ASN.1 to define its APDUs. This is so because the protocol specification can simply import relevant types and values from the abstract service specification.

b) The concrete realization of an abstract service whose abstract operations do not report their outcomes is conceptually simple. This is so because each such abstract operation represents an interaction comprising a single APDU. From this simplest of all possible interactions, arbitrarily complex ones can be constructed.

11 Proprietary realizations

A secondary objective of CCITT Recommendations and ISO Standards is to ensure that those portions of a distributed information processing task that are carried out by proprietary means are accomplished in such a way that the intended overall functionality of the system is upheld.

The realization of abstract ports and bindings by proprietary means is briefly discussed below. Both distributed and non-distributed realizations are considered.

11.1 *Distributed realizations*

The concrete realization of ports and bindings by means of proprietary computer communication protocols is a local matter. The specification of the visible functionality embodied in the abstract service provides a guide to the implementors of the proprietary realizations, so that, where such realizations are appropriate, they may play the appropriate role in the overall task.

11.2 *Non-distributed realizations*

The concrete realization of ports and bindings by means of mechanisms wholly within a single computer is a local matter. As with the case considered in § 11.1, the abstract service specification serves as a guide to the implementor in ensuring that the proprietary realization can nonetheless play the appropriate role in the overall task.

ANNEX A

(to Recommendation X.407)

Example of use of abstract service notation

This Annex is not a part of this Recommendation.

This Annex illustrates the use of the abstract model and service notation by means of an example. The example involves two systems, the *Yellow* and *Green Systems*, and their environments, the **Yellow** and **Green Environments**.

It uses the abstract model notation to describe the environments separately (§ A.2 and A.4) and to show how their systems are related: one is constructed from the other (§ A.6). It uses the abstract service notation to describe the capabilities of each system (§ A.3 and A.5). The example concludes by realizing the systems' ports as ACs and ASEs using the ROS notation of Recommendation X.219, as might be appropriate for OSI communication (§ A.7 and A.8).

A.1 *Assignment of object identifiers*

The ASN.1 modules defined in this Annex require the assignment of a variety of Object Identifiers. All are defined below using ASN.1. The assignments are definitive except for those for ASN.1 modules and the subject of application service definition conventions itself. The definitive assignments for the former occur in the modules themselves; other references to them appear in IMPORT clauses. The latter is fixed.

```
ExampleObjectIdentifiers {joint-iso-ccitt
        mhs-motis(6) asdc(2) example(1) modules(0) object-identifiers(0)}
DEFINITIONS IMPLICIT TAGS ::=
BEGIN
-- Prologue
-- Exports everything.

IMPORTS -- nothing -- ;

ID ::= OBJECT IDENTIFIER

-- Abstract Service Definition Conventions Example (not definitive)

id-asdc-ex ID ::= {joint-iso-ccitt mhs-motis (6) asdc(2) example(1)}
        -- not definitive

-- Categories

id-mod  ID ::= {id-asdc-ex 0} -- modules; not definitive
id-ot   ID ::= {id-asdc-ex 1} -- object types
id-pt   ID ::= {id-asdc-ex 2} -- port types
id-ref  ID ::= {id-asdc-ex 3} -- refinements
id-ac   ID ::= {id-asdc-ex 4} -- application contexts
id-ase  ID ::= {id-asdc-ex 5} -- application service elements
id-as   ID ::= {id-asdc-ex 6} -- abstract syntaxes

-- Modules

id-mod-object-identifiers  ID ::= {id-mod 0} -- not definitive
id-mod-ye-refinement       ID ::= {id-mod 1} -- not definitive
id-mod-y-abstract-service  ID ::= {id-mod 2} -- not definitive
id-mod-ge-refinement       ID ::= {id-mod 3} -- not definitive
id-mod-g-abstract-service  ID ::= {id-mod 4} -- not definitive
id-mod-ys-refinement       ID ::= {id-mod 5} -- not definitive
id-mod-ys-realization      ID ::= {id-mod 6} -- not definitive
id-mod-gs-realization      ID ::= {id-mod 7} -- not definitive

-- Object types

id-ot-y-environment     ID ::= {id-ot 0}
id-ot-y-user            ID ::= {id-ot 1}
id-ot-y-system          ID ::= {id-ot 2}
id-ot-g-environment     ID ::= {id-ot 3}
id-ot-g-user            ID ::= {id-ot 4}
id-ot-g-manager         ID ::= {id-ot 5}
id-ot-g-system          ID ::= {id-ot 6}
id-ot-agent             ID ::= {id-ot 7}

-- Port types

id-pt-y-use ID          ID ::= {id-pt 0}
id-pt-g-use ID          ID ::= {id-pt 1}
id-pt-g-management      ID ::= {id-pt 2}

-- Refinements

id-ref-y-environment    ID ::= {id-ref 0}
id-ref-g-environment    ID ::= {id-ref 1}
id-ref-y-system         ID ::= {id-ref 2}

-- Application contexts

id-ac-y-use             ID ::= {id-ac 0}
id-ac-g-use             ID ::= {id-ac 1}
id-ac-g-management      ID ::= {id-ac 2}
```

-- *Application service elements*

```
id-ase-y-use              ID ::= { id-ase 0 }
id-ase-g-use              ID ::= { id-ase 1 }
id-ase-g-management       ID ::= { id-ase 2 }
```

-- *Abstract syntaxes*

```
id-as-y-use               ID ::= { id-as 0 }
id-as-g-use               ID ::= { id-as 1 }
id-as-g-management        ID ::= { id-as 2 }
```

END -- *of ExampleObjectIdentifiers*

A.2 *Refinement of yellow environment*

The Yellow Environment, depicted in Figure A-1/X.407, is formally refined below using the OBJECT and REFINE macros.

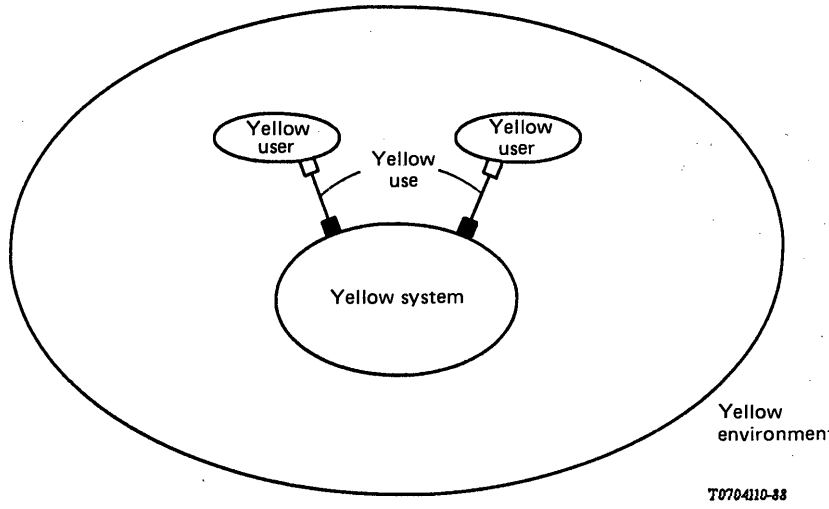


FIGURE A-1/X.407

The yellow environment

As Figure A-1/X.407 indicates and the ASN.1 specification below confirms, the Yellow Environment can be modeled as an object which can be decomposed into one central object, the **Yellow System**, and any number of other, peripheral objects, **yellow users**. The Yellow System interacts with yellow users by means of its **yellow-use** ports.

```
YellowEnvironmentRefinement { joint-iso-ccitt
      mhs-motis(6) asdc(2) example(1) modules(0) ye-refinement(1) }
DEFINITIONS IMPLICIT TAGS ::=
BEGIN
-- Prologue

EXPORTS
      yellow-environment, yellow-environment-refinement,
      yellow-system, yellow-user;
```

```
IMPORTS
        -- Yellow Abstract Service
                yellow-use
                ....
                FROM YellowAbstractService {joint-iso-ccitt
                        mhs-motis(6) asdc(2) example(1) modules(0) y-abstract-service(2)}

        -- Example Object Identifiers
                id-ot-y-environment, id-ot-y-system, id-ot-y-user,
                id-ref-y-environment
                ....
                FROM ExampleObjectIdentifiers {joint-iso-ccitt
                        mhs-motis(6) asdc(2) example(1) modules(0) object-identifiers(0)}

        -- Abstract Service Notation
                OBJECT, REFINE
                ....
                FROM AbstractServiceNotation {joint-iso-ccitt
                        mhs-motis(6) asdc(2) modules(0) notation(1)};


-- Yellow Environment

yellow-environment OBJECT
        ::= id-ot-y-environment


-- Yellow Environment refinement

yellow-environment-refinement REFINE yellow-environment AS
        yellow-user RECURRING
        yellow-system
                yellow-use [S] PAIRED WITH yellow-user
        ::= id-ref-y-environment


-- Component object types

yellow-user OBJECT
        PORTS {
                yellow-use [C]}
        ::= id-ot-y-user

yellow-system OBJECT
        PORTS {
                yellow-use [S]}
        ::= id-ot-y-system

END -- of YellowEnvironmentRefinement
```

A.3 *Definition of yellow abstract service*

The abstract service that the Yellow System provides to its users is formally defined below using the PORT and ABSTRACT-BIND, -OPERATION, and -ERROR macros.

As the ASN.1 specification indicates, the abstract service that the Yellow System provides comprises ports of a single kind, yellow-use. Each port comprises a number of abstract operations which collectively report a number of abstract errors. The Yellow System guards its ports by means of an abstract bind operation, **YellowBind**, which demands that users identify themselves convincingly before further interaction occurs. An abstract unbind operation, **YellowUnbind**, which constitutes the finalization step required to conclude an interaction.

```
YellowAbstractService {joint-iso-ccitt
        mhs-motis(6) asdc(2) example(1) modules(0) y-abstract-service(2)}
DEFINITIONS IMPLICIT TAGS ::=
BEGIN
-- Prologue
```

```
EXPORTS
        AuthenticateUser, Yellow-operation-1, ... yellow-use;

IMPORTS
        -- Example Object Identifiers
                id-pt-y-use
                ....
                FROM ExampleObjectIdentifiers { joint-iso-ccitt
                        mhs-motis(6) asdc(2) example(1) modules(0) object-identifiers(0) }
        -- Abstract Service Notation
                ABSTRACT-BIND, ABSTRACT-ERROR, ABSTRACT-OPERATION, PORT
                ....
                FROM AbstractServiceNotation { joint-iso-ccitt
                        mhs-motis(6) asdc(2) modules(0) notation(1) };


-- Port type

yellow-use PORT
        CONSUMER INVOKES {
                Yellow-operation-1, ... }
        ::= id-pt-y-use


-- Abstract bind operation

Credentials ::= SET {
                name     [0] IA5String,
                password [1] IA5String }

YellowBind ::= ABSTRACT-BIND
                TO { yellow-use[S] }
        BIND
                ARGUMENT credentials Credentials
                        BIND-ERROR ENUMERATED {
                                name-or-password-invalid(0) }


-- Abstract unbind operation

YellowUnbind ::= ABSTRACT-UNBIND
        FROM { yellow-use[S] }


-- Abstract operations

Yellow-operation-1 ::= ABSTRACT-OPERATION
                ARGUMENT ...
                RESULT ...
                ERRORS {
                        yellow-error-1, ... }
...


-- Abstract errors

yellow-error-1 ABSTRACT-ERROR
                PARAMETER ...
                ::= 1
...

END -- of YellowAbstractService
```

A.4 *Refinement of green environment*

The Green Environment, depicted in Figure A-2/X.407, is formally refined below using the OBJECT and REFINE macros.

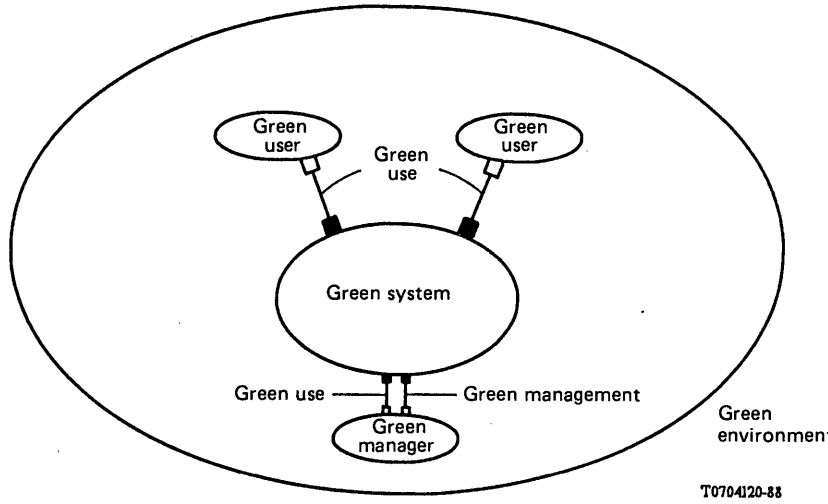


FIGURE A-2/X.407

The green environment

As Figure A-2/X.407 indicates and the ASN.1 specification below confirms, the Green Environment can be modeled as an object which can be decomposed into one central object, the **Green System**; any number of other, peripheral objects, **green users**; and any number of yet additional objects, **green managers**. The Green System interacts with green users and managers by means of its **green-use** ports, and with green managers (alone) by means of its **green-management** ports.

```
GreenEnvironmentRefinement {joint-iso-ccitt
      mhs-motis(6) asdc(2) example(1) modules(0) ge-refinement(3)}
DEFINITIONS IMPLICIT TAGS ::=
BEGIN
-- Prologue


EXPORTS
      green-environment, green-environment-refinement,
      green-manager, green-system, green-user;


IMPORTS
      -- Green Abstract Service
            green-use, green-management
            ....
            FROM GreenAbstractService {joint-iso-ccitt
                  mhs-motis(6) asdc(2) example(1) modules(0) g-abstract-service(4)}

      -- Example Object Identifiers
            id-ot-g-environment, id-ot-g-manager, id-ref-g-environment
            id-ot-g-user, id-ot-g-system,
            ....
            FROM ExampleObjectIdentifiers {joint-iso-ccitt
                  mhs-motis(6) asdc(2) example(1) modules(0) object-identifiers(0)}

      -- Abstract Service Notation
            OBJECT, REFINE
            ....
            FROM AbstractServiceNotation {joint-iso-ccitt
                  mhs-motis(6) asdc(2) modules(0) notation(1)};
```

```
-- Green Environment

green-environment OBJECT
        ::= id-ot-g-environment


-- Green Environment refinement

green-environment-refinement REFINE green-environment AS
        green-user      RECURRING
        green-manager RECURRING
        green-system
                green-use               [S] PAIRED WITH green-user, green-manager
                green-management [S] PAIRED WITH green-manager
        ::= id-ref-g-environment


-- Component object types

green-user OBJECT
        PORTS {
                green-use               [C] }
        ::= id-ot-g-user

green-manager OBJECT
        PORTS {
                green-use               [C],
                green-management [C] }
        ::= id-ot-g-manager

green-system OBJECT
        PORTS {
                green-use               [S],
                green-management [S] }
        ::= id-ot-g-system

END -- of GreenEnvironmentRefinement
```

A.5 *Definition of green abstract service*

The abstract service that the Green System provides to its users and managers is formally defined below using the PORT and ABSTRACT-BIND, -OPERATION, and -ERROR macros.

As the ASN.1 specification indicates, the abstract service that the Green System provides comprises ports of two kinds, green-use and green-management. A port of either kind comprises a number of abstract operations which collectively report a number of abstract errors. The Green System guards its ports by means of abstract bind operations, **AuthenticateUser** and **AuthenticateManager**, which demand that users and managers identify themselves convincingly before further interaction can occur. No abstract unbind operations are specified, indicating that no finalization step is required to conclude an interaction.

```
GreenAbstractService { joint-iso-ccitt
        mhs-motis(6) asdc(2) example(1) modules(0) g-abstract-service(4) }
DEFINITIONS IMPLICIT TAGS ::=
BEGIN
-- Prologue


EXPORTS
        AuthenticateManager, AuthenticateUser, green-management,
        Green-management-operation-1, ... green-use,
        Green-use-operation-1, ...;
```

```
IMPORTS
        -- Example Object Identifiers
                id-pt-g-use, id-pt-g-management
                ....
                FROM ExampleObjectIdentifiers { joint-iso-ccitt
                        mhs-motis(6) asdc(2) example(1) modules(0) object-identifiers(0) }

        -- Abstract Service Notation
                PORT, ABSTRACT-BIND, ABSTRACT-OPERATION, ABSTRACT-ERROR
                ....
                FROM AbstractServiceNotation { joint-iso-ccitt
                        mhs-motis(6) asdc(2) modules(0) notation(1) };


-- Port types

green-use PORT
        CONSUMER INVOKES {
                Green-use-operation-1, ... }
        ::= id-pt-g-use

green-management PORT
        CONSUMER INVOKES {
                Green-management-operation-1, ... }
        ::= id-pt-g-management


-- Abstract bind operations

Credentials ::= SET {
        name     [0] IA5String,
        password [1] IA5String }

AuthenticateUser ::= ABSTRACT-BIND
        ARGUMENT credentials Credentials
        BIND-ERROR ENUMERATED {
                name-or-password-invalid(0) }

AuthenticateManager ::= ABSTRACT-BIND
        ARGUMENT credentials Credentials
        BIND-ERROR ENUMERATED {
                name-or-password-invalid(0),
                not-a-manager           (1) }


-- Abstract operations

Green-use-operation-1 ::= ABSTRACT-OPERATION
        ARGUMENT ...
        RESULT ...
        ERRORS {
                green-error-1, ... }
...

Green-management-operation-1 ::= ABSTRACT-OPERATION
        ARGUMENT ...
        RESULT ...
        ERRORS {
                green-error-1, ... }
...


-- Abstract errors

green-error-1 ABSTRACT-ERROR
        PARAMETER ...
        ::= 1
...

END -- of GreenAbstractService
```

A.6 *Refinement of yellow system*

The Yellow System, depicted in Figure A-3/X.407, is formally refined below using the OBJECT and REFINE macros.

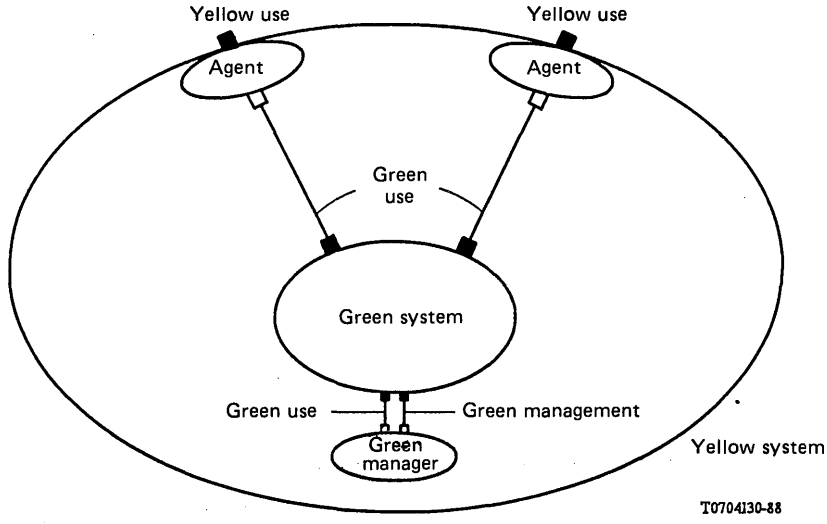


FIGURE A-3/X.407

The yellow system

As the figure indicates and the ASN.1 specification confirms, the Yellow System, when examined closely, has components. In particular, the Yellow System comprises the Green System and green managers, augmented by objects of an as yet unseen variety, *agent.* An **agent** serves as an intermediary between the Green System and a yellow user. It might be thought of as adding value to the Green System. In any case, it is a provider of a yellow-use port and a consumer of a green-use port.

```
YellowSystemRefinement { joint-iso-ccitt
      mhs-motis(6) asdc(2) example(1) modules(0) ys-refinement(5) }
DEFINITIONS IMPLICIT TAGS ::=
BEGIN
-- Prologue


EXPORTS
      agent, yellow-system-refinement;
```

```
IMPORTS
        -- Yellow Environment Refinement
                yellow-system, yellow-use
                ....
                FROM YellowEnvironmentRefinement {joint-iso-ccitt
                        mhs-motis(6) asdc(2) example(1) modules(0) ye-refinement(1)}

        -- Green Environment Refinement
                green-management, green-manager, green-system,
                green-use
                ....
                FROM GreenEnvironmentRefinement {joint-iso-ccitt
                        mhs-motis(6) asdc(2) example(1) modules(0) ge-refinement(3)}

        -- Example Object Identifiers
                id-ot-agent, id-ref-y-system
                ....
                FROM ExampleObjectIdentifiers {joint-iso-ccitt
                        mhs-motis(6) asdc(2) example(1) modules(0) object-identifiers(0)}

        -- Abstract Service Notation
        OBJECT, REFINE
                FROM AbstractServiceNotation {joint-iso-ccitt
                        mhs-motis(6) asdc(2) modules(0) notation(1)};


-- Yellow System refinement

yellow-system-refinement REFINE yellow-system AS
        agent RECURRING
                yellow-use          [S] VISIBLE
        green-manager RECURRING
        green-system
                green-use           [S] PAIRED WITH agent, green-manager
                green-management    [S] PAIRED WITH green-manager
        ::= id-ref-y-system


-- Component object type

agent OBJECT
        PORTS {
                yellow-use          [S],
                green-use           [C]}
        ::= id-ot-agent

END -- of YellowSystemRefinement
```

A.7 *Realization of yellow system*

The abstract service of the Yellow System is formally realized below, by means of ROS, using the APPLICATION-CONTEXT and APPLICATION-SERVICE-ELEMENT macros of Recommendation X.219.

As the ASN.1 specification indicates, the abstract service that the Yellow System provides is realized as a single ASE, **yellow-use-ASE**, and a single and corresponding AC, **yellow-use-AC**. Each abstract bind operation, abstract operation, or abstract error in the abstract service has a corresponding and equivalent bind operation, operation, or error, respectively, in its ROS-based realization.

Note that Integer values are assigned to the operations; the corresponding abstract operations require and received no such values.

```
YellowSystemRealization {joint-iso-ccitt
        mhs-motis(6) asdc(2) example(1) modules(0) ys-realization(6)}
DEFINITIONS IMPLICIT TAGS ::=
BEGIN
-- Prologue
```

```
EXPORTS
        yellow-use-AC, yellow-use-ASE;

IMPORTS
        -- Yellow Abstract Service
                Yellow-operation-1, ... yellow-use, YellowBind,
                YellowUnbind
                ....
                FROM YellowAbstractService { joint-iso-ccitt
                        mhs-motis(6) asdc(2) example(1) modules(0) y-abstract-service(2) }

        -- Example Object Identifiers
                id-ac-y-use, id-as-y-use, id-ase-y-use
                ....
                FROM ExampleObjectIdentifiers { joint-iso-ccitt
                        mhs-motis(6) asdc(2) example(1) modules(0) object-identifiers(0) }

        -- Remote Operations APDUs
                rOSE
                ....
                FROM Remote-Operations-APDUs { joint-iso-ccitt
                        remote-operations(4) apdus(1) }

        -- Association Control
                aCSE
                ....
                FROM Remote-Operation-Notation-extension
                        { joint-iso-ccitt remote-operations(4)
                        notation-extension(2) }

        -- Remote Operations Notation Extension
                APPLICATION-CONTEXT, APPLICATION-SERVICE-ELEMENT
                ....
                FROM Remote-Operations-Notation-extension { joint-iso-ccitt
                        remote-operations(4) notation-extension(2) };

ACSE-AS OBJECT IDENTIFIER ::=
                { joint-iso-ccitt association-control(2)
                abstractSyntax(1) apdus(0) version1(1) }

-- Application context

yellow-use-AC APPLICATION-CONTEXT
        APPLICATION SERVICE ELEMENTS { aCSE }
        BIND YellowBind
        UNBIND YellowUnbind
        REMOTE OPERATIONS { rOSE }
        INITIATOR CONSUMER OF { yellow-use-ASE }
        ABSTRACT SYNTAXES { yellow-use-AS, aCSE-AS }
        ::= id-ac-y-use

-- Application service element

yellow-use-ASE APPLICATION-SERVICE-ELEMENT
        CONSUMER INVOKES {
                yellow-operation-1, ... }
        ::= id-ase-y-use

yellow-operation-1 Yellow-operation-1 ::= 1
...

-- Abstract syntax

yellow-use-AS OBJECT IDENTIFIER ::= id-as-y-use

END -- of YellowSystemRealization
```

A.8 *Realization of green system*

The abstract service of the Green System is formally realized below, by means of ROS, using the APPLICATION-CONTEXT and APPLICATION-SERVICE-ELEMENT macros of Recommendation X.219.

As the ASN.1 specification indicates, the abstract service that the Green System provides is realized as two ASEs, **green-use-ASE** and **green-management-ASE**, and two, corresponding ACs, **green-use-AC** and **green-management-AC**. Each abstract bind operation, abstract operation, or abstract error in the abstract service has a corresponding and equivalent bind operation, operation, or error, respectively, in its ROS-based realization.

Note that Integer values are assigned to the operations; the corresponding abstract operations require and received no such values.

```
GreenSystemRealization {joint-iso-ccitt
    mhs-motis(6) asdc(2) example(1) modules(0) gs-realization(7)}
DEFINITIONS IMPLICIT TAGS ::=
BEGIN
-- Prologue

EXPORTS
    green-management-AC, green-management-ASE, green-use-AC,
    green-use-ASE;

IMPORTS
    -- Green Abstract Service
        AuthenticateManager, AuthenticateUser,
        green-management, Green-management-operation-1, ...
        green-use, Green-use-operation-1, ...
        ....
        FROM GreenAbstractService {joint-iso-ccitt
            mhs-motis(6) asdc(2) example(1) modules(0) g-abstract-service(4)}

    -- Example Object Identifiers
        id-ac-g-use,          id-ase-g-use,          id-as-g-use,
        id-ac-g-management,   id-ase-g-management,   id-as-g-management
        ....
        FROM ExampleObjectIdentifiers {joint-iso-ccitt
            mhs-motis(6) asdc(2) example(1) modules(0) object-identifiers(0)}

    -- Remote Operations APDUs
        rOSE
        ....
        FROM Remote-Operations-APDUs {joint-iso-ccitt
            remote-operations(4) apdus(1)}

    -- Association Control
        aCSE
        ....
        FROM Remote-Operations-Notation-extension {joint-iso-ccitt
            remote-operations(4) notation-extension(2)}

    -- Remote Operations Notation Extension
        APPLICATION-CONTEXT, APPLICATION-SERVICE-ELEMENT
        ....
        FROM Remote-Operations-Notation-extension {joint-iso-ccitt
            remote-operations(4) notation-extension(2)};

aCSE-AS OBJECT IDENTIFIER ::=
        {joint-iso-ccitt association-control(2)
        abstractSyntax(1) apdus(0) version1(1)}
```

```
-- Application contexts

green-use-AC APPLICATION-CONTEXT
        APPLICATION SERVICE ELEMENTS { aCSE }
        BIND AuthenticateUser
        UNBIND NoOperation
        REMOTE OPERATIONS { rOSE }
        INITIATOR CONSUMER OF { green-use-ASE }
        ABSTRACT SYNTAXES { green-use-AS, aCSE-AS }
        ::= id-ac-g-use

green-management-AC APPLICATION-CONTEXT
        APPLICATION SERVICE ELEMENTS { aCSE }
        BIND AuthenticateManager
        UNBIND NoOperation
        REMOTE OPERATIONS { rOSE }
        INITIATOR CONSUMER OF { green-management-ASE }
        ABSTRACT SYNTAXES { green-management-AS, aCSE-AS }
        ::= id-ac-g-management

NoOperation ::= UNBIND


-- Application service elements

green-use-ASE APPLICATION-SERVICE-ELEMENT
        CONSUMER INVOKES {
                green-use-operation-1, ... }
        ::= id-ase-g-use

green-management-ASE APPLICATION-SERVICE-ELEMENT
        CONSUMER INVOKES {
                green-management-operation-1, ...}
        ::= id-ase-g-management

green-use-operation-1 Green-use-operation-1 ::= 1
...

green-management-operation-1 Green-management-operation-1 ::= 50 ...

-- Abstract syntaxes

green-use-AS OBJECT IDENTIFIER ::= id-as-g-use

green-management-AS OBJECT IDENTIFIER ::= id-as-g-management

END -- of GreenSystemRealization
```

ANNEX B

(to Recommendation X.407)

Reference definition of object identifiers

This Annex is an integral part of this Recommendation.

This Annex defines for reference purposes various Object Identifiers cited in the ASN.1 modules of Annex C. It uses ASN.1.

With the exception of those assigned in Annex A, all Object Identifiers this Recommendation assigns are assigned in this Annex. The Annex is definitive for all but those for ASN.1 modules and the subject of application service definition conventions itself. The definitive assignments for the former occur in the modules themselves; other references to them appear in IMPORT clauses. The latter is fixed.

```
ASDCObjectIdentifiers { joint-iso-ccitt
        mhs-motis(6) asdc(2) modules(0) object-identifiers(0) }
DEFINITIONS IMPLICIT TAGS ::=
BEGIN
-- Prologue

-- Exports everything.

IMPORTS -- nothing--;

ID ::= OBJECT IDENTIFIER

-- Abstract Service Definition Conventions (not definitive)

id-asdc ID ::= { joint-iso-ccitt mhs-motis(6) asdc(2) } -- not definitive

-- Categories

id-mod ID ::= { id-asdc 0 } -- modules; not definitive
id-ex  ID ::= { id-asdc 1 } -- example; not definitive

-- Modules

id-mod-object-identifiers ID ::= { id-mod 0 } -- not definitive
id-mod-notation           ID ::= { id-mod 1 } -- not definitive

END -- of ASDCObjectIdentifiers
```

ANNEX C

(to Recommendation X.407)

Reference definition of notation

This Annex is an integral part of this Recommendation.

This Annex, a supplement to section 2, defines for reference purposes the notation for specifying abstract models and services. It employs ASN.1.

```
AbstractServiceNotation { joint-iso-ccitt
        mhs-motis(6) asdc(2) modules(0) notation(1) }
DEFINITIONS IMPLICIT TAGS ::=
BEGIN
-- Prologue

EXPORTS
        ABSTRACT-BIND, ABSTRACT-ERROR, ABSTRACT-OPERATION,
        ABSTRACT-UNBIND, OBJECT, PORT, REFINE;

IMPORTS
        -- Remote Operations Notation
                BIND, ERROR, OPERATION, UNBIND
                ....
                FROM Remote-Operation-Notation { joint-iso-ccitt
                        remote-operations(4) notation(0) };
```

```
-- Object macro

OBJECT; MACRO ::=
BEGIN

TYPE NOTATION          ::= "PORTS" "{" PortList "}" | empty
VALUE NOTATION         ::= value (VALUE OBJECT IDENTIFIER)

PortList               ::= Port " , " PortList | Port
Port                   ::= value (PORT) PortType

PortType               ::= Symmetric | Asymmetric

Symmetric              ::= empty
Asymmetric             ::= Consumer | Supplier

Consumer               ::= "[C]"
Supplier               ::= "[S]"

END

-- Port macro

PORT MACRO ::=
BEGIN

TYPE NOTATION          ::= Operations | empty
VALUE NOTATION         ::= value (VALUE OBJECT IDENTIFIER)

Operations             ::= Symmetrical | Asymmetrical

Symmetrical            ::= "ABSTRACT" "OPERATIONS" "{" OperationList "}"
Asymmetrical           ::= OneSided | TwoSided

OneSided               ::= Consumer | Supplier
TwoSided               ::= Consumer Supplier | Supplier Consumer

Consumer               ::= "CONSUMER" "INVOKES" "{" OperationList "}"
Supplier               ::= "SUPPLIER" "INVOKES" "{" OperationList "}"

OperationList          ::= Operation " , " OperationList | Operation
Operation              ::= value (ABSTRACT-OPERATION)

END

-- Refine macro

REFINE MACRO ::=
BEGIN

TYPE NOTATION          ::= Object "AS" ComponentList
VALUE NOTATION         ::= value (VALUE OBJECT IDENTIFIER)

ComponentList          ::= Component ComponentList | Component
Component              ::= ObjectSpec PortSpecList

ObjectSpec             ::= Object | Object "RECURRING"

PortSpecList           ::= PortSpec PortSpecList | PortSpec
PortSpec               ::= value (PORT) PortType PortStatus

PortType               ::= Consumer | Supplier | empty
Consumer               ::= "[C]"
Supplier               ::= "[S]"

PortStatus             ::= "VISIBLE" | "PAIRED" "WITH" ObjectList

ObjectList             ::= Object "," ObjectList | Object
Object                 ::= value (OBJECT)

END
```

```
-- Abstract bind, unbind, operation, and error macros

ABSTRACT-BIND MACRO ::=
BEGIN

TYPE NOTATION       ::= Ports Bind
VALUE NOTATION      ::= value (VALUE BindType)

Ports               ::= "TO" "{" PortList "}" | empty
PortList            ::= Port " , " PortList | Port
Port                ::= value (PORT) PortSide
PortSide            ::= Consumer | Supplier | empty
Consumer            ::= "[C]"
Supplier            ::= "[S]"

Bind                ::= type (BindType) -- must be a BIND type
                        | empty <BindType ::= BIND>

END

ABSTRACT-UNBIND MACRO ::=
BEGIN

TYPE NOTATION       ::= Ports Unbind
VALUE NOTATION      ::= value (VALUE UnbindType)

Ports               ::= "FROM" "{" PortList "}" | empty
PortList            ::= Port " , " PortList | Port
Port                ::= value (PORT) PortSide
PortSide            ::= Consumer | Supplier | empty
Consumer            ::= "[C]"
Supplier            ::= "[S]"

Unbind              ::= type (UnbindType) -- must be an UNBIND type
                        | empty <UnbindType ::= UNBIND>

END

ABSTRACT-OPERATION MACRO ::= OPERATION

ABSTRACT-ERROR MACRO ::= ERROR

END -- of AbstractServiceNotation
```

ANNEX D

(to Recommendation X.407)

Differences between CCITT Recommendation and ISO Standard

This Annex is not a part of this Recommendation.

This Annex lists all but the purely stylistic differences between this Recommendation and the corresponding ISO International Standard.

No differences between the two specifications exist.

ANNEX E

(to Recommendation X.407)

Index

This Annex indexes this Recommendation. It gives the number(s) of the section(s) on which each item in each of several categories is defined. Its coverage of each category is exhaustive.

This Annex indexes items (if any) in the following categories:

a) abbreviations;
b) terms;
c) information items;
d) ASN.1 modules;
e) ASN.1 macros;
f) ASN.1 types;
g) ASN.1 values;
h) bilateral agreements;
i) items for further study;
j) items to be supplied.

E.1 *Abbreviations*

E.2 *Terms*

MESSAGE HANDLING SYSTEMS: ENCODED INFORMATION TYPE CONVERSION RULES

(Malaga-Torremolinos, 1984; amended at Melbourne, 1988)

The establishment in various countries of telematic services and computer-based store-and-forward message services in association with public data networks creates a need to produce standards to facilitate international message exchange between subscribers to such services.

The CCITT,

considering

(a) the need for interpersonal messaging and message transfer services;

(b) the need to transfer messages of different types having a large variety of formats;

(c) that the F-series of Recommendations defines telematic services;

(d) that the T-series of Recommendations defines terminal equipment and control procedures for telematic services;

(e) that the V-series of Recommendations provides the means for data communication over the telephone network;

(f) that Recommendation X.200 defines the reference model of open systems interconnection for CCITT applications;

(g) that a set of Recommendations describes various aspects of message handling X.400, X.402, X.403, X.407, X.408, X.411, X.413, X.419 and X.420;

(h) that the CCITT and ISO have drawn up a suitable set of convertion rules;

(i) that, for specific user applications, some variations may be developed and applied by bilateral agreement,

unanimously declares the view

(1) that the purpose and scope of this Recommendation are described in § 1;

(2) that general aspects of the rules for converting between encoded information types are described in § 2;

(3) that conversion rules for particular encoded information types are defined in subsequent sections.

TABLE OF CONTENTS

3 *Conversion from TLX*

- 3.1 Conversion from TLX to IA5Text
- 3.2 Conversion from TLX to TTX
- 3.3 Conversion from TLX to G3Fax
- 3.4 Conversion from TLX to G4Class1
- 3.5 Conversion from TLX to Videotex
- 3.6 Conversion from TLX to voice
- 3.7 Conversion from TLX to mixedmode

4 *Conversion from IA5Text*

- 4.1 Conversion from IA5Text to TLX
- 4.2 Conversion from IA5Text to TTX
- 4.3 Conversion from IA5Text to G3Fax
- 4.4 Conversion from IA5Text to G4Class1
- 4.5 Conversion from IA5Text to Videotex
- 4.6 Conversion from IA5Text to voice
- 4.7 Conversion from IA5Text to mixedmode

5 *Conversion from TTX*

- 5.1 Conversion from TTX to TLX
- 5.2 Conversion from TTX to IA5Text
- 5.3 Conversion from TTX to TTX
- 5.4 Conversion from TTX to G3Fax
- 5.5 Conversion from TTX to G4Class1
- 5.6 Conversion from TTX to Videotex
- 5.7 Conversion from TTX to voice
- 5.8 Conversion from TTX to mixedmode

6 *Conversion from G3Fax*

- 6.1 Conversion from G3Fax to G3Fax
- 6.2 Conversion from G3Fax to G4Class1
- 6.3 Conversion from G3Fax to mixedmode

7 *Conversion from G4Class1*

- 7.1 Conversion from G4Class1 to G3Fax
- 7.2 Conversion from G4Class1 to G4Class1
- 7.3 Conversion from G4Class1 to mixedmode

8 *Conversion from Videotex*

- 8.1 Conversion from Videotex to TLX
- 8.2 Conversion from Videotex to IA5Text
- 8.3 Conversion from Videotex to TTX
- 8.4 Conversion from Videotex to G3Fax
- 8.5 Conversion from Videotex to G4Class1
- 8.6 Conversion from Videotex to Videotex
- 8.7 Conversion from Videotex to voice
- 8.8 Conversion from Videotex to mixedmode

9 *Conversion from Voice*

- 9.1 Conversion from voice to voice.

1 Purpose and scope

This is one of a set of Recommendations for Message Handling. The entire set provides a comprenhensive specification for Message Handling comprising any number of cooperating open systems.

This Recommendation specifies the algorithms the MHS uses when converting between different types of encoded information.

Other aspects of Message Handling are defined in other Recommendations. The overall system and services of Message Handling is specified in Recommendation X.400. The overall architecture of the MHS is defined in Recommendation X.402. The conformance testing of MHS components is described in Recommendation X.403. The conventions used in the definition of the abstract services provided by MHS components are defined in Recommendation X.407. The abstract service the MTS provides and the procedures that govern its distributed operation are defined in Recommendation X.411. The abstract service the MS provides is defined in Recommendation X.413. The application protocols that govern the interaction of MHS components are specified in Recommendation X.419. The interpersonal Messaging System, an application of Message Handling, is specified in Recommendation. X.420.

Paragraph 2 of this Recommendation describes the general aspects of converion for the MHS. In §§ 3 to 10, particular conversion rules are defined. Annex A provides code conversion tables. Annex B lists the abbreviations used.

2 General aspects of conversion

Among the data elements subject to conversion are the subject and body of a message. The conversion has two aspects, format and code. The aspect of control is described elsewhere in the relevant Recommendations.

2.1 *Encoded information types*

This Recommendation defines the conversion rules for eight types of encoded information utilized in the MHS. In order to refer to each types, the following terms are used:

TLX	The code is defined in Recommendation F.1. The format is defined in Recommendation S 5.
IA5Text	The code is defined in Recommendation T.50.
TTX	The format is defined in Recommendations F.200 and T.60, and the code is defined in Recommendation T.61.
G3Fax	The encoding scheme is defined in Recommendation T.4, and the signalling of the encoding scheme is defined in Recommendation T.30.
G4Class1	The format and encoding scheme are defined in Recommendations T.6, T.503 and T.563.
Videotex	The format and encoding scheme are defined in Recommendations T.101, T.504 and T.541.
Voice	The encoding scheme is for further study.
Mixedmode	The format and encoding scheme are defined in Recommendations T.501 and T.561.

Note – TLX: Telex, TTX: Teletex.

The TTX and G3Fax types have two subtypes: basic and optional. The G4Class1 type and the mixedmode type have two subtypes: basic and non-basic.

Table 1/X.408 depicts all conceivable conversions between the above types and subtypes. It characterizes each as: (–) No conversion, (a) possible without loss of information, (b) possible but loss of information may occur, or (c) impractical. This Recommendation defines the rules for format and code conversion for conversions in the second and third categories.

TABLE 1/X.408

Encoded information type conversions

From \ To		TLX	IA5 Text	TTX		G3 Fax		G4 Class 1		Videotex	Voice	Mixed mode	
				basic	optional [1]	basic	optional [1]	basic	non basic [1]			basic	non basic [1]
TLX [4]		–	b [7]	a	a	a	a	a	a	b	FS	a	a
IA5 Text		b	–	b	b	b	b	b	b	b	FS	b	b
TTX	basic	b	b	–	a	a	a	a	a	a	FS	a	a
	optional [1]	b	b	b	b [2], [3]	b	b	b	b	a	FS	a	b [2], [3]
G3 Fax	basic	c	c	c	c	–	a	a	a	c [5]	c	a	a
	optional [1]	c	c	c	c	b	b [2], [3]	b	b	c [5]	c	b	b
G4 Class 1	basic	c	c	c	c	b	b	–	a	c [5]	c	a	a
	non basic [1]	c	c	c	c	b	b	b	b [2], [3]	c [5]	c	b	b [2], [3]
Videotex		b	b	b	b	b [6]	b [6]	b [6]	b [6]	b	FS	FS	FS
Voice		c	c	c	c	c	c	c	c	c	FS	c	c
Mixed mode	basic	b	b	b	b	b	b	a	a	b	FS	–	a
	non basic [1]	b	b	b	b [2], [3]	b	b	b	b [2], [3]	b	FS	b	b [2], [3]

– No conversion
a Possible without loss of information
b Possible but loss of information may occur
c Impractical
FS For further study

[1] Specified in the relevant Recommendations.

[2] No information is lost if the originating and recipient terminals have the same optional functions.

[3] Informations may be lost if the originating terminal uses optional functions that the recipient terminal lacks.

[4] The WHO ARE YOU character is assumed to be a protocol element used for communicating with the Telex terminal and not part of the message's content.

[5] It may be possible with loss of information, if the recipient terminal has the capability of the photographic type of information.

[6] When converting videotex, color information may be lost.

[7] In the case when IA5 Text has less than 69 characters available in a line, format informations may be lost.

2.2 *Two aspects of conversion*

The conversion rules have two aspects:

1) the format aspect;
2) the code aspect.

The conversion rules for encoded information types which have logical structures are for further study.

2.2.1 *Ground rules*

If there is an existing standard on the conversion between different types, it should be referred to without any modifications unless required. If there is not, the following ground rules are specified:

1) If there are standards on the subject and object types, the conversion rules should be defined such that the intersecting part of the standards is preserved. The creation of new rules for non-intersecting parts should be based on clear requirements, otherwise they should not be created.
2) When either the subject or object type has no standard, the conversion rules should be defined such that standard types can be accommodated as much as possible for both directions of the conversion.
3) When neither type has a standard, the definition of the rules is for further study.

2.2.2 *Format aspect*

The format aspect represents the dimensional attributes of the presentation space of user messages.

The two-dimensional (X and Y) aspect of the conversion is to be specified for a message being transferred. Following are the parameters to be defined for this aspect. Whether voice should be considered in the same context is for further study.

a) The X-direction of the presentation space is defined by means of either:
   1) the size of a character and the number of characters to be presented;
   2) length.

   If the object type has a smaller size of the X-direction than the subject type, a mechanism for adjustment of line length, such as the insertion of a CR/LF pair, should be defined as the format conversion rule.

b) The Y-direction of the presentation space is defined by means of either:
   1) the number of lines per presentation space or per unit length;
   2) length.

   If the object type has a smaller page size of the Y-direction than the subject type, a mechanism for change of page format, such as the insertion of a CR/FF pair, should be defined as the format conversion rule. If the object type has no length limitation of Y-direction and the subject type is pagenated, some format conversion rule, such as the insertion of one or more blank lines, should be defined to represent the page boundary.

   When converting characters to G3Fax or G4Class1, the rules for imaging should be applied according to Recommendation T.351.

2.2.3 *Code aspect*

With respect to the code aspect, Annex A specifies the conversion between different types. Further notes can be found in each subsection if necessary.

2.3 *Loss of information*

2.3.1 *Initial assumption on loss of information*

When considering conversion between different encoded information types some initial assumptions were taken into account. Changes to the character font, character size or paper type, etc. are not regarded as loss of information.

2.3.2 *Format loss of information*

The format conversions take two different forms; line length and number of lines. The following definitions apply:

1) *Line length (number of characters)*
   a) originator's line length less than, or equal to recipient's line length: no loss of information;
   b) originator's line length more than recipient's line length: loss of information.
2) *Page length (number of lines)*
   a) originator's page length less than, or equal to recipient's page length: no loss of information. However, in the recipient's pages, a clean field should be inserted between the originator's pages;
   b) originator's page length more than recipient's page: no loss of information, provided the originator's page is mapped onto an integer number of destination pages.

Note – Information may also be lost due to differences between the printable and reproducible areas in facsimile.

2.3.3 *Code loss of information*

If a graphic character is reproduced identically on both systems then there is no loss of information. However, if there is a change between italic, bold, underlined, normal or coloured (as in the case of Videotex) this could be considered to be a loss of information. (e.g., one type of rendition could mean a positive financial result and the other a negative result!). This requires further study.

A conversion to a similar character, but not identical, in the second system is loss of information. A conversion from one character to many characters (e.g., $ to dollar) is also loss of information.

2.4 *Encoded information type properties*

Followings are the properties of encoded information types assumed for this Recommendation.

2.4.1 *TLX encoded information type*

The TLX encoded information type is not paginated. A TLX text line contains maximum 69 graphic characters as defined in Recommendation S.5. The end of a line is also represented by an ITA2 CR-LF pair.

2.4.2 *IA5Text encoded information type*

The IA5Text encoded information type is paginated. An IA5Text line contains maximum N_1 graphic characters. The end of a line is also represented by an IA5 CR-LF pair. An IA5Text page contains maximum M_1 lines. The end of a page is also represented by an IA5 CR-FF pair.

Note – Any value to N_1 or M_1 are not assurmed in this Recommendation unless explicitly specified in the relevant sections. A common value to N_1 is 80.

2.4.3 *TTX encoded information type*

The TTX encoded information type is paginated. The format of a TTX page is as defined by following "default condition of basic Teletex":

- paper size and orientation: vertical basic page format;
- character spacing: 2.54 mm;
- line feed spacing: 4.23 mm;
- rendition: default rendition.

This implies that the maximum number of characters per line is 77 and the maximum number of lines per page is 55. The end of a TTX line is represented by a TTX CR-LF pair. The end of a TTX page is represented by a TTX CR-FF pair.

Note – When converting from Teletex, each Teletex line shall be preceded by 5 spaces reduced by the number of backspaces (BS) found in the beginning of each Teletex line (refer to Recommendation F.200, § 7.6.9.1).

2.4.4 *G3Fax encoded information type*

The G3Fax encoded information type is paginated. The format of a G3Fax is described in Recommendation T.4.

2.4.5 *G4Class1 encoded information type*

The G4Class1 encoded information type is paginated. The format of a G4Class1 is described in Recommendation T.563.

2.4.6 *Videotex encoded information type*

The Videotex encoded information type is paginated. A Videotex line contains maximum N_2 graphic characters. The end of a line is represented by a Videotex APD-APR pair. An Videotex page contains maximum M_2 lines in the defined-diplay area. The end of a page is represented by an Videotex CS. The function of scroll is not assumed.

Among various graphic elements that Videotex can handle, the following rules apply only to the alphanumeric characters in the display-data elements. The use of Interworking Data Syntax (IDS) requires further study.

Note – Values of N_2 or M_2 are not assumed in this Recommendation; they may differ according to the syntax used.

2.4.7 *Voice encoded information type*

Requires further study.

2.4.8 *Mixedmode encoded information type*

The format and encoding scheme are defined in Recommendations T.501 and T.561.

3 Conversion from TLX

3.1 *Conversion from TLX to IA5Text*

3.1.1 *Format conversion*

A TLX line is directly converted into an IA5Text line if the number of graphic characters of the code-converted TLX line does not exceed the maximum number of graphic characters available in an IA5Text line (N_1) and if the number of lines in a TLX does not exceeds the number of lines available in an IA5Text page (M_1).

An ITA2 CR-LF pair invokes a new IA5Text line. If an LF is not associated with a CR, an LF may be inserted after the CR.

A TLX line should be converted into the appropriate number of IA5Text lines (with possible insertion of an IA5 CR-LF pair) if the number of graphic characters of the code-converted TLX line exceeds N_1. Each split IA5Text line (except the last one) may contain the maximum number of graphic characters available for the IA5Text line. Folding at the word boundary requires further study.

A TLX should be split into the appropriate number of IA5Text pages if the number of code-converted TLX line exceeds M_1. Each split IA5Text page (except the last one) may contain the maximum number of lines available for the IA5Text page. The number of lines in a TLX should be calculated after the insertion of any required IA5 CR-LF pairs.

Note – Inclusion of data escape mode in a TLX is for further study.

Those aspects other than above (e.g., character spacing, line spacing and so on) are outside the scope of this conversion rule.

3.1.2 *Code conversion*

This conversion rule is defined in Annex A.

3.2 *Conversion from TLX to TTX*

3.2.1 *Format conversion*

A TLX text is directly converted into a TTX text if the number of graphic characters of the code converted TLX line does not exceed the maximum number of graphic characters available in a TTX text line and if the number of lines in a TLX does not exceeds the number of lines available in a TTX page.

An ITA2 CR-LF pair invokes a new TTX text line. If an LF is not associated with a CR, an LF may be inserted after the CR.

A TLX text line should be converted into the appropriate number of TTX text lines (with possible insertion of a TTX CR-LF pair) if the number of graphic characters in a code-converted TLX line exceeds the maximum number of graphic characters available in a TTX text line. Each split TTX text line (except the last one) may contain the maximum number of graphic characters available for the TTX text line. Folding at the word boundary requires further study.

A TLX text should be split into the appropriate number of TTX pages (with possible insertion of a TTX CR-FF pair) if the number of code-converted TLX line exceeds the number available in a TTX page. Each split TTX page (except the last one) may contain the maximum number of lines available for the TTX page. The number of lines in a TLX should be calculated after the insertion of any required TTX CR-LF pairs.

3.2.2 *Code conversion*

This conversion rule is defined in Annex A.

3.3 *Conversion from TLX to G3Fax*

3.3.1 *Format conversion*

A TLX text is directly converted into a G3Fax if the number of graphic characters of the code-converted TLX line does not exceed the maximum number of graphic characters available in a G3Fax character line and if the number of lines in the TLX does not exceeds the number of lines available in a G3Fax pag.

An ITA2 CR-LF pair invokes a new G3Fax character line. If an LF is not associated with an CR, a LF may be inserted after the CR.

A TLX text line should be converted into the appropriate number of G3Fax character lines (with possible insertion of an ITA2 CR-LF pair) if the number of graphic characters in a code-converted TLX line exceeds the maximum number of graphic characters available in a G3Fax character line. Each split G3Fax character line (except the last one) may contain the maximum number of graphic characters available for the G3Fax character line. Folding at the word boundary requires further study.

A TLX text should be split into the appropriate number of G3Fax pages if the number of code-converted TLX lines exceeds the number available in a G3Fax page. Each split G3Fax page (except the last one) may contain the maximum number of lines available for the G3Fax page. The number of lines in a TLX should be calculated after the insertion of any required ITA2 CR-LF pairs.

Imaging of characters to G3Fax should be in accordance with Recommendation T.351.

3.3.2 *Code conversion*

This conversion rule is defined in Annex A. The charater rendition is a national option.

3.4 *Conversion from TLX to G4Class1*

3.4.1 *Format conversion*

A TLX text is directly converted into a G4Class1 if the number of graphic characters of the code-converted TLX line does not exceed the maximum number of graphic characters available in a G4Class1 character line and if the number of lines in the TLX does not exceeds the number of lines available in a G4Class1 page.

An ITA2 CR-LF pair invokes a new G4Class1 character line. If an LF is not associated with a CR, an LF may be inserted after the CR.

A TLX text line should be converted into the appropriate number of G4Class1 character lines (with possible insertion of a ITA2 CR-LF pair) if the number of graphic characters in a code-converted TLX line exceeds the maximum number of graphic characters available in a G4Class1 character line. Each split G4Class1 character line (except the last one) may contain the maximum number of graphic characters available for the G4Class1 character line. Folding at the word boundary requires further study.

A TLX text should be split into the appropriate number of G4Class1 pages if the number of code-converted TLX lines exceeds the number available in a G4Class1 page. Each split G4Class1 page (except the last one) may contain the maximum number of lines available for the G4Class1 page. The number of lines in a TLX should be calculated after the insertion of any required ITA1 CR-LF pairs.

Imaging of characters to G4Class1 should be in accordance with Recommendation T.351.

3.4.2 *Code conversion*

This conversion rule is defined in Annex A. The character rendition is a national option.

3.5 *Conversion from TLX to Videotex*

3.5.1 *Format conversion*

A TLX text is converted directly into a Videotex if the number of graphic characters of the code-converted TLX line does not exceed the number of charcters available in the Videotex line, and if the number of lines in the TLX does not exceed the number of lines available in the Videotex page.

An ITA2 CR-LF pair invokes a new Videotex line. If an LF is not associated with a CR, an LF may be inserted after the CR.

A TLX text line should be converted into the appropriate number of Videotex lines (with possible insertion of a T.101 APD-APR pair) if the number of graphic characters in a code-converted TLX line exceeds the maximum number of graphic charcters available in a Videotex line (N_2). Each split Videotex line (except the last one) may contain the maximum number of graphic characters available for the Videotex line. Folding at the word boundary requires further study.

A TLX text should be split into the appropriate number of Videotex pages if the number of code-converted TLX lines exceeds the number available in a Videotex page (M_2). Each split Videotex page (except the last one) may contain the maximum number of lines available for the Videotex page. The number of lines in a TLX should be calculated after the insertion of any required T.101 APD-APR pairs.

By definition, if one TLX line is converted into one Videotex line and the TLX text is converted into one or several Videotex pages, this is not considered as loss of information.

3.5.2 *Code conversion*

The code conversion is specified in Annex A.

3.6 *Conversion from TLX to voice*

Requires further study.

3.7 *Conversion from TLX to mixedmode*

Requires further study.

4 Conversion from IA5Text

4.1 *Conversion from IA5Text to TLX*

4.1.1 *Format conversion*

An IA5Text is converted directly into a TLX if the number of graphic characters of the code-converted IA5Text line does not exceed the number of characters available in the TLX line.

An IA5 CR-LF pair invokes a new TLX line. If an LF is not associated with a CR, an LF may be inserted after the CR.

An IA5Text line should be converted into the appropriate number of TLX lines (with possible insertion of an IA5 CR-LF pair) if the number of graphic characters in a code-converted TLX line exceeds the maximum number of graphic characters available in a TLX line. Each split TLX line (except the last one) may contain the maximum number of graphic characters available for the TLX line. Folding at the word boundary requires further study.

An IA5 CR-FF pair is converted into an ITA2 CR-LF pair plus the optional addition of up to 3 blank lines.

4.1.2 *Code conversion*

This conversion rule is defined in Annex A.

When the IA5Text changes from a letter to a figure type then an ITA2 Figure Shift shall be generated. When the IA5Text changes from a figure to a letter then an ITA2 Letter Shift shall be generated.

At the start of the message an ITA2 Letter Shift shall be generated to ensure the TLX is in a known shift mode.

4.2 *Conversion from IA5Text to TTX*

4.2.1 *Format conversion*

An IA5Text is directly converted into a TTX if the number of graphic characters of the code-converted IA5Text line does not exceed the maximum number of graphic characters available in a TTX line and if the number of lines in the IA5Text does not exceeds the number of lines available in a TTX page.

An IA5 CR-LF pair invokes a new TTX line. If an LF is not associated with a CR, an LF may be inserted after the CR.

An IA5Text line should be converted into the appropriate number of TTX lines (with possible insertion of a TTX CR-LF pair) if the number of graphic characters in a code-converted IA5Text line exceeds the maximum number of graphic characters available in a TTX line. Each split TTX line (except the last one) may contain the maximum number of graphic characters available for the TTX line. Folding at the word boundary requires further study.

An IA5 CR-FF pair invokes a new TTX page.

An IA5Text page should be split into the appropriate number of TTX pages (with possible insertion of TTX CR-FF pair) if the number of code-converted IA5Text lines exceeds the number available in a TTX page. Each split TTX page (except the last one) may contain the maximum number of lines available for the TTX page. The number of lines in an IA5Text should be calculated after the insertion of any required TTX CR-LF pairs.

4.2.2 *Code conversion*

Every IA5 character is represented by seven bits (b_7-b_1). Characters of IA5 are converted into the corresponding characters of T.61 by adding 0 as the eighth bit (b_8). The conversion rules is specified in Annex A.

Note – In the case of circumflex accent, grave accent, and overline, whether diacritical marks of T.61 can be chosen as converted codes is for further study.

4.3 *Conversion from IA5Text to G3Fax*

4.3.1 *Format conversion*

An IA5Text is directly converted into a G3Fax if the number of graphic characters of the code-converted IA5Text line does not exceed the maximum number of graphic characters available in a G3Fax character line and if the number of lines in the IA5Text does not exceed the number of lines available in a G3Fax page.

An IA5 CR-LF pair invokes a new G3Fax character line. If an LF is not associated with a CR, an LF may be inserted after the CR.

An IA5Text line should be converted into the appropriate number of G3Fax character lines (with possible insertion of an IA5 CR-LF pair) if the number of graphic characters in a code-converted IA5Text line exceeds the maximum number of graphic characters available in a G3Fax character line. Each split G3Fax character line (except the last one) may contain the maximum number of graphic characters available for the G3Fax character line. Folding at the word boundary requires further study.

An IA5 CR-FF pair invokes a new G3Fax page.

An IA5Text page should be split into the appropriate number of G3Fax pages if the number of code-converted IA5Text lines exceeds the number available in a G3Fax page. Each split G3Fax page (except the last one) may contain the maximum number of lines available for the G3Fax page. The number of lines in an IA5Text should be calculated after the insertion of any required IA5 CR-LF pairs.

When converting from IA5Text to G3Fax, the G3Fax image format will be 80 characters per line with a left marging of 20 mm and 55 lines per page.

Imaging of characters to G3Fax should be in accordance with Recommendation T.351.

4.3.2 *Code conversion*

This conversion rule is defined in Annex A. The character rendition is a national option.

4.4 *Conversion from IA5Text to G4Class1*

4.4.1 *Format conversion*

An IA5Text is directly converted into a G4Class1 if the number of graphic characters of the code-converted IA5Text line does not exceed the maximum number of graphic characters available in a G4Class1 character line and if the number of lines in the IA4Text does not exceeds the number of lines available in a G4Class1 page.

An IA5 CR-LF pair invokes a new G4Class1 character line. If an LF is not associated with a CR, an LF may be inserted after the CR.

An IA5Text line should be converted into the appropriate number of G4Class1 character lines (with possible insertion of an IA5 CR-LF pair) if the number of graphic characters in a code-converted IA5Text line exceeds the maximum number of graphic characters available in a G4Class1 character line. Each split G4Class1 character line (except the last one) may contain the maximum number of graphic characters available for the G4Class1 character line. Folding at the word boundary requires further study.

An IA5 CR-FF pair invokes a new G4Class1 page.

An IA5Text page should be split into the appropriate number of G4Class1 pages if the number of code-converted IA5Text lines exceeds the number available in a G4Class1 page. Each split G4Class1 page (except the last one) may contain the maximum number of lines available for the G4Class1 page. The number of lines in an IA5Text should be calculated after the insertion of any required IA5 CR-LF pairs.

When converting from IA5Text to G4Class1, the G4Class1 image format will be 80 characters per line with a left margin of 20 mm and 55 lines per page.

Imaging of characters to G4Class1 should be in accordance with Recommendation T.351.

4.4.2 *Code conversion*

This conversion rule is defined in Annex A. The character rendition is a national option.

4.5 *Conversion from IA5Text to Videotex*

4.5.1 *Format conversion*

An IA5Text is directly converted into a Videotex if the number of graphic characters of the code-converted IA5Text line does not exceed the maximum number of graphic characters available in a Videotex line and if the number of lines in the IA5Text does not exceeds the number of lines available in a Videotex page.

An IA5 CR-LF pair invokes a new Videotex line. If an LF is not associated with a CR, an LF may be inserted after the CR.

An IAa5Text line should be converted into the appropriate number of Videotex lines (with possible insertion of a T.101 APD-APR pair) if the number of graphic characters in a code-converted IA5Text line exceeds the maximum number of graphic characters available in a Videotex line. Each split Videotex character line (except the last one) may contain the maximum number of graphic characters available for the Videotex character line. Folding at the word boundary requires further study.

An IA5 CR-FF pair invokes a new Videotex page.

An IA5Text page should be split into the appropriate number of Videotex pages (with possible insertion of a T.101 CS) if the number of code-converted IA5Text lines exceeds the number available in a Videotex page. Each split Videotex page (except the last one) may contain the maximum number of lines available for the Videotex page. The number of lines in a IA5Text should be calculated after the insertion of any required IA5 CR-LF pairs.

By definition, if one IA5Text line is converted into one Videotex line and if a IA5Text page is converted into one or several Videotex pages, each IA5Text page starting with a new Videotex page, this is not considered as loss of information.

4.5.2 *Code conversion*

The conversion rule is specified in Annex A.

4.6 *Conversion from IA5Text to voice*

Requires further study.

4.7 *Conversion from IA5Text to mixedmode*

Requires further study.

5 Conversion from TTX

5.1 *Conversion from TTX to TLX*

5.1.1 *Format conversion*

A TTX is converted directly into a TLX if the number of graphic characters of the code-converted TTX line does not exceed the number of characters available in the TLX line.

A TTX CR-LF pair invokes a new TLX line. If an LF is not associated with a CR, an LF may be inserted after the CR.

A TTX line should be converted into the appropriate number of TLX lines (with possible insertion of an ITA2 CR-LF pair) if the number of graphic characters in a code-converted TLX line exceeds the maximum number of graphic characters available in a TLX line. Each split TLX line (except the last one) may contain the maximum number of graphic characters available for the TLX line. Folding at the word boundary requires further study.

A TTX CR-FF pair is converted into an ITA2 CR-LF pair plus the optional addition of up to 3 blank lines.

5.1.2 *Code conversion*

This conversion rule is defined in Annex A.

When the TTX changes from a letter to a figure type then an ITA2 Figure Shift shall be generated. When the TTX changes from a figure to a letter then an ITA2 Letter Shift shall be generated.

At the start of the message an ITA2 Letter Shift shall be generated to ensure the TLX is in a known shift mode.

5.2 *Conversion from TTX to IA5Text*

5.2.1 *Format conversion*

A TTX is converted into an IA5Text assuming the vertical orientation and a maximum of 77 characters per line (a line may be constructed by placing 72 characters to the right of the left marging and additional 5 characters to the left of the left margin). Teletex information in the horizontal orientation will result in loss of information.

A TTX is directly converted into an IA5Text if the number of graphic characters of the code-converted TTX line does not exceed the maximum number of graphic characters available in a IA5Text character line and if the number of lines in the TTX does not exceed the number of lines available in a IA5Text page.

A TTX CR-LF pair invokes a new IA5Text line. If an LF is not associated with an CR, an LF may be inserted after the CR.

A TTX line should be converted into the appropriate number of IA5Text lines (with possible insertion of an IA5 CR-LF pair) if the number of graphic characters in a code-converted TTX line exceeds the maximum number of graphic characters available in an IA5Text line. Each split IA5Text character line (except the last one) may contain the maximum number of graphic characters available for the IA5Text character line. Folding at the word boundary requires further study.

A TTX CR-FF pair invokes a new IA5Text page.

A TTX page should be split into the appropriate number of IA5Text pages if the number of code-converted TTX lines exceeds the number available in a IA5Text page. Each split IA5Text page (except the last one) may contain the maximum number of lines available for the IA5Text page. The number of lines in a TTX should be calculated after the insertion of any required IA5 CR-LF pairs.

5.2.2 *Code conversion*

Every character in the set of T.61 is converted into the corresponding character of IA5 by deleting Bit b_8. The conversion rule is specified in Annex A.

Note – Other conversion rules for the currency sings are for further study.

5.3 *Conversion from TTX to TTX*

Requires further study.

5.4 *Conversion from TTX to G3Fax*

5.4.1 *Format conversion*

A TTX text is directly converted into a G3Fax if the number of graphic characters of the code-converted TTX line does not exceed the maximum number of graphic characters available in a G3Fax character line and if the number of lines in the TTX does not exceeds the number of lines available in a G3Fax page.

A TTX CR-LF pair invokes a new G3Fax character line. If an LF is not associated with a CR, an LF may be inserted after the CR.

A TTX line should be converted into the appropriate number of G3Fax character lines (with possible insertion of an TTX CR-LF pair) if the number of graphic characters in a code-converted TTX line exceeds the maximum number of graphic characters available in a G3Fax character line. Each split G3Fax character line (except the last one) may contain the maximum number of graphic characters available for the G3Fax character line. Folding at the word boundary requires further study.

A TTX CR-FF pair invokes a new G3Fax page.

A TTX page should be split into the appropriate number of G3Fax pages (with possible insertion of a TTX CR-FF pair) if the number of code-converted TTX lines exceeds the number available in a G3Fax page. Each split G3Fax page (except the last one) may contain the maximum number of lines available for the G3Fax page. The number of lines in an TTX should be calculated after the insertion of any required TTX CR-LF pairs.

Imaging of characters to G3Fax should be in accordance with Recommendation T.351. The use of figures, however, in Recommendation T.351 corresponding to the options of TTX requires further study.

5.4.2 *Code conversion*

This conversion rule is defined in Annex A. The character rendition is a national option.

5.5 *Conversion from TTX to G4Class1*

5.5.1 *Format conversion*

A TTX text is directly converted into a G4Class1 if the number of graphic characters of the code-converted TTX line does not exceed the maximum number of graphic characters available in a G4Class1 character line and if the number of lines in the TTX does not exceeds the number of lines available in a G4Class1 page.

A TTX CR-LF pair invokes a new G4Class1 line. If an LF is not associated with a CR, an LF may be inserted after the CR.

A TTX line should be converted into the appropriate number of G4Class1 character lines (with possible insertion of a TTX CR-LF pair) if the number of graphic characters in a code-converted TTX line exceeds the maximum number of graphic characters available in a G4Class1 character line. Each split G4Class1 character line (except the last one) may contain the maximum number of graphic characters available for the Videotex character line. Folding at the word boundary requires further study.

A TTX CR-FF pair invokes a new G4Class1 page.

A TTX page should be split into the appropriate number of G4Class1 pages (with possible insertion of a TTX CR-FF pair) if the number of code-converted TTX lines exceeds the number available in a G4Class1 page. Each split G4Class1 page. Each split G4Class1 page (except the last one) may contain the maximum number of lines available for the G4Class1 page. The number of lines in a TTX should be calculated after the insertion of any required TTX CR-LF pairs.

Imaging of characters to G4Class1 should be in accordance with Recommendation T.351. The use of figures, however, in Recommendation T.351 corresponding to the options of TTX requires further study.

5.5.2 *Code conversion*

The conversion rule is specified in Annex A. The character rendition is a national option.

5.6 *Conversion from TTX to Videotex*

5.6.1 *Format conversion*

A TTX is converted into an Videotex assuming the vertical orientation and a maximum of 77 characters per line (a line may be constructed by placing 72 characters to the right of the left margin and additional 5 characters to the left of the left margin). Teletex information in the horizontal orientation will result in loss of information.

Note – BS at the beginning (maximum 5) of a TTX line moves the first logical character position of all Videotex lines to the left, according to the number of BS. This enables an extention the line length by 5 characters. An appropriate number of spaces has to be added at the beginning of each Videotex line in order to ensure that all Videotex lines begin at the first logical position given by the line having the most BS at the beginning of the TTX line.

A TTX is directly converted into an Videotex if the number of graphic characters of the code-converted TTX line does not exceed the maximum number of graphic characters available in a Videotex line and if the number of lines in the TTX does not exceed the number of lines available in a Videotex page.

A TTX CR-LF pair invokes a new Videotex line. If an LF is not associated with a CR, an LF may be inserted after the CR.

A TTX line should be converted into the appropriate number of Videotex lines (with possible insertion of an T.101 APD-APR pair) if the number of graphic characters in a code-converted TTX line exceeds the maximum number of graphic characters available in an Videotex line. Each split Videotex character line (except the last one) may contain the maximum number of graphic characters available for the Videotex character line. Folding at the word boundary requires further study.

A TTX CR-FF pair invokes a new Videotex page.

A TTX page should be split into the appropriate number of Videotex pages (with possible insertion of a T.101 CS) if the number of code-converted TTX lines exceeds the number available in a Videotex page. Each split Videotex page (except the last one) may contain the maximum number of lines available for the Videotex page. The number of lines in a TTX should be calculated after the insertion of any required IA5 CR-LF pairs.

By definition, if one TTX line is converted into one Videotex line and if a TTX page is converted into one or several Videotex pages, each TTX page starting with a new Videotex page, this is not considered as loss of information.

5.6.2 *Code conversion*

The conversion rule is specified in Annex A.

5.7 *Conversion from TTX to voice*

Requires further study.

5.8 *Conversion from TTX to mixedmode*

5.8.1 *Format conversion*

Requires further study.

5.8.2 *Code conversion*

Not required. T.61 String is allowed in mixedmode.

6 Conversion from G3Fax

6.1 *Conversion from G3Fax to G3Fax*

Requires further study.

6.2 *Conversion from G3Fax to G4Class1*

Requires further study.

6.3 *Conversion from G3Fax to mixedmode*

Requires further study.

7 Conversion from G4Class1

7.1 *Conversion from G4Class1 to G3Fax*

Requires further study.

7.2 *Conversion from G4Class1 to G4Class1*

Requires further study.

7.3 *Conversion from G4Class1 to mixedmode*

Requires further study.

8 Conversion from Videotex

8.1 *Conversion from Videotex to TLX*

8.1.1 *Format conversion*

A Videotex is directly converted into a TLX if the number of graphic characters of the code-converted Videotex line does not exceed the maximum number of graphic characters available in a TLX line.

A T.101 APD-APR pair invokes a new TLX line. If an APR is not associated with an APD, an APR may be inserted after the APD.

A Videotex line should be converted into the appropriate number of TLX lines (with possible insertion of an ITA2 CR-LF pair) if the number of graphic characters in a code-converted Videotex line exceeds the maximum number of graphic characters available in a TLX line. Each split TLX line (except the last one) may contain the maximum number of graphic characters available for the TLX line. Folding at the word boundary requires further study.

A T.101 CS is converted into an ITA2 CR-LF pair plus the optional addition of up to 3 blank lines.

8.1.2 *Code conversion*

This conversion rule is defined in Annex A.

8.2 *Conversion from Videotex to IA5Text*

8.2.1 *Format conversion*

A Videotex is directly converted into an IA5Text if the number of graphic characters of the code-converted Videotex line does not exceed the maximum number of graphic characters available in a IA5Text line and if the number of lines in the Videotex does not exceed the number of lines available in a IA5Text page.

A T.101 APD-APR pair invokes a new IA5Text line. If an APR is not associated with an APD, an APR may be inserted after the APR.

A Videotex line should be converted into the appropriate number of IA5Text lines (with possible insertion of an IA5 CR-LF pair) if the number of graphic characters in a code-converted Videotex line exceeds the maximum number of graphic characters available in a IA5Text line. Each split IA5Text line (except the last one) may contain the maximum number of graphic characters available for the IA5Text line. Folding at the word boundary requires further study.

A T.101 CS invokes a new IA5Text page or, alternatively, 3 blank lines if the next videotex page can be represented fully on the same IA5Text page.

A Videotex page should be split into the appropriate number of IA5Text pages (with possible insertion of a IA5 CR-FF pair) if the number of code-converted Videotex lines exceeds the number available in a IA5Text page. Each split IA5Text page (except the last one) may contain the maximum number of lines available for the IA5Text page. The number of lines in an Videotex should be calculated after the insertion of any required IA5 CR-LF pairs.

By definition, if one Videotex line is converted into one IA5Text line and if multiple Videotex pages are converted into one IA5Text pages, each IA5Text page starting with a new Videotex page, this is not considered as loss of information.

8.2.2 *Code conversion*

The conversion rule is specified in Annex A.

8.3 *Conversion from Videotex to TTX*

8.3.1 *Format conversion*

A Videotex is directly converted into a TTX if the number of graphic characters of the code-converted Videotex line does not exceed the maximum number of graphic characters available in a TTX line and if the number of lines in the Videotex does not exceed the number of lines available in a TTX page.

A T.101 APD-APR pair invokes a new TTX line. If an APR is not associated with an APD, an APR may be inserted after the APR.

A Videotex line should be converted into the appropriate number of TTX lines (with possible insertion of a TTX CR-LF pair) if the number of graphic characters in a code-converted Videotex line exceeds the maximum number of graphic characters available in a TTX line. Each split TTX line (except the last one) may contain the maximum number of graphic characters available for the TTX line. Folding at the word boundary requires further study.

A T.101 CS invokes a new TTX page or, alternatively, 3 blank lines if the next videotex page can be represented fully on the same TTX page.

A Videotex page should be split into the appropriate number of TTX pages (with possible insertion of a TTX CR-FF pair) if the number of code-converted Videotex lines exceeds the number available in a TTX page. Each split TTX page (except the last one) may contain the maximum number of lines available for the TTX page. The number of lines in an Videotex should be calculated after the insertion of any required TTX CR-LF pairs.

By definition, if one Videotex line is converted into one TTX line and if multiple Videotex pages are converted into one TTX pages, each TTX page starting with a new Videotex page, this is not considered as loss of information.

8.3.2 *Code conversion*

The conversion rule is specified in Annex A.

8.4 *Conversion from Videotex to G3Fax*

Requires further study.

8.5 *Conversion from Videotex to G4Class1*

Requires further study.

8.6 *Conversion from Videotex to Videotex*

Requires further study.

8.7 *Conversion from Videotex to voice*

Requires further study.

8.8 *Conversion from Videotex to mixedmode*

Requires further study.

9 Conversion from voice

9.1 *Conversion from voice to voice*

Requires further study.

10 Conversion from mixedmode

10.1 *Conversion from mixedmode to TLX*

Requires further study.

10.2 *Conversion from mixedmode to IA5Text*

Requires further study.

10.3 *Conversion from mixedmode to TTX*

Requires further study.

10.4 *Conversion from mixedmode to G3Fax*

Requires further study.

10.5 *Conversion from mixedmode to G4Class1*

Requires further study.

10.6 *Conversion from mixedmode to Videotex*

Requires further study.

10.7 *Conversion from mixedmode to voice*

Requires further study.

10.8 *Conversion from mixedmode to mixedmode*

Requires further study.

ANNEX A

(to Recommendation X.408)

Code conversion tables

A.1 *Introduction*

This Annex was developed for describing the code conversion rules for the use of message handling concisely and consistently.

A.2 *Premises*

A.2.1 *References*

a) Tables 1/S.18 and 2/S.18 (Rules);

b) Tables 1/T.50 to 9/T.50 and 11/T.50 (Symbol and description);

c) Figures 1/T.51 and 2/T.51, Tables 1/T.51 and 2/T.51 and Tables 4/T.51 and 5/T.51 (Symbol and description);

d) Table C-1/T.60 (Rules);

e) Paragraphs 3.2 and 3.3 of T.61 (Symbol and identification);

f) Figures 2/T.61 and 3/T.61, Tables 1/T.61 and 2/T.61 and Figures B-1/T.61 and C-1/T.61 (Symbol)

g) T.100 (Identification);

h) X.408 (1984) (Rules)

A.2.2 *Structure of the tables*

A.2.2.1 *Introduction*

The tables are divided into two columns:

a) REFERENCE SET

b) CONVERTED SET

A pair of the REFERENCE SET and a sub-column in the CONVERTED SET column form the definition of the code conversion rule for the encoded information type referenced as an output.

The tables are developed under the premises described below. The extension of the tables requires further study.

A.2.2.2 *REFERENCE SET*

The REFERENCE SET is a collection of the final visible graphic form (e.g. printed or displayed) of characters. This set is NOT intended to introduce a new character set to be implemented elsewhere.

Regarding control characters, the visibility of the character could not form a criterion to enlist it to the REFERENCE SET. Any available controls are enlisted in order to show if the intent of the particular control character is maintained after conversion.

Note – Use of a control character may be different from one type to other. This may imply that we need to deal with character escape sequences defined in ISO 2022 first, then character conversion. This requires further study.

This column is provided for the reference character set. The set is a conceptual one and may contain any conceivable characters. It is completely independent of the peculiar encoding of individual characters. There are, however, three exceptions on the characters "circumflex accent", "grave accent" and "overline/tilde" of IA5 because of historical and technical reasons.

The symbols # and ¤ whose encoding is different from one type to another are assigned the same identification numbers.

Questions relating to the registration and maintenance authority of the REFERENCE SET are left for further study.

The column has three sub-columns:

a) *Identification:* Identification number for a character increased by 10 and also the identification code developed in T.61 if available. Numbers 0 through 999 will be allocated for the controls and numbers 1000 or over will be allocated to graphic characters.

b) *Name or description:* Concise description of a character.

c) *Symbol:* Known symbol for a character.

A.2.2.3 *Column "CONVERTED SET"*

This column has a number of sub-columns. Each sub-column defines a corresponding character(s) to that of the REFERENCE SET.

Some part of the conversion rule may not necessarily be used when converting one type to the other. For example, in the Telex-to-Teletex conversion, there is no need to use the part for the characters numbered more than 2000.

Originally, five sub-columns are provided for:

a) Teletex encoded information type (referred to as T.61);

b) IA5 text encoded information type (referred to as IA5IRV);

c) Telex encoded information type (referred to as ITA2);

d) G3Fax and G4Class1 encoded information type (referred as facsimile);

e) Videotex encoded information type (referred as Videotex).

A.2.3 *Use of other standards*

For message handling applications, the conversion table shall be used wherever possible. The conversion table does not attempt to replace existing international standards that deal with conversions for other applications other than message handling.

The basic rules for the use of the conversion tables shall be that the conversions defined in the tables are derived from the basic character representations of the codes as found in the appropriate international standards. For this version of X.408, alternative representations may be used but are for further study.

When there are alternative conversions defined in existing Recommendations, X.408 shall not constrain choices. Examples of alternative conversion are as follows:

a) ITA2 → IA5 conversion defined in Recommendation S.18 has alternative conversions such as:
The IA5 Asterisk character * can be converted to either ITA2 character ? or characters (?).

b) ITA2 → T.61 conversion defined in Recommendation T.60 has alternative conversions such as:
The ITA2 character A can be converted to either T.61 character A or a.

Note that the conversion from one graphic character to many graphic characters may be used where appropriate as a national option.

A.3 *Conventions*

a) If both SYM and No. boxes are left blank in the "CONVERTED SET" column, this means that the conversion is not provided (i.e., NOT convertible).

b) FS means "for further study".

c) disc. means "DISCARDED".

d) The symbols for the control characters should not be interpreted as to print the symbols literally. They are simply placed for reference information. The intent is that semantics of those characters defined elsewhere should be kept. See § A.2.2.2.

A.4 *Notes*

ⓐ This character is used only to operate the answer-back unit for corresponding equipment in the international public service.

ⓑ These characters have no international allocation.

ⓒ These characters have no corresponding function in other encoded information types. Conversion equipment operates the appropriate shift and discard the characters.

ⓓ Classification of control functions numbered less than 1000 requires further study.

TABLE A-1/X.408

REFERENCE SET				CONVERTED SET									
Identification		Name or description [d]	Symbol	T.61		IA5IRV		ITA2		Facsimile		Videotex	
No.				Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.
0		Null	NUL	disc.		NUL	0	?	1310	disc.		NUL	0
10		Start of heading	SOH	disc.		SOH	10	?	1310	disc.		disc.	
20		Start of text	STX	disc.		STX	20	?	1310	disc.		disc.	
30		End of text	ETX	disc.		ETX	30	?	1310	disc.		disc.	
40		End of transmission	EOT	disc.		EOT	40	?	1310	disc.		disc.	
50		Enquiry	ENQ	disc.		ENQ	50	WRU [a]	600	disc.		ENQ	50
60		Acknowledge	ACK	disc.		ACK	60	?	1310	disc.		disc.	
70		Bell	BEL	disc.		BEL	70	BEL	70	disc.		disc.	
80	CF10	Backspace	BS	BS	80	BS	80	?	1310	BS	80	APB	800
90		Horizontal tabulation	HT	disc.		HT	90	?	1310	disc.		disc.	
100	CF12	Line feed	LF	LF	100	LF	100	LF	100	LF	100	APD	820
110		Vertical tabulation	VT	disc.		VT	110	?	1310	disc.		disc.	
120	CF14	Form feed	FF	CR,FF	130,120	FF	120	CR,LF	130,100	CR,FF	130,120	CS	840
130	CF15	Carriage return	CR	CR	130	CR	130	CR	130	CR	130	APR	850
140		Shift-out	SO	disc.		SO	140	?	1310	disc.		FS	
150		Shift-in	SI	disc.		SI	150	?	1310	disc.		FS	
160		Data link escape	DLE	disc.		DLE	160	?	1310	disc.		disc.	
170		Device control one	DC1	disc.		DC1	170	?	1310	disc.		disc.	
180		Device control two	DC2	disc.		DC2	180	?	1310	disc.		disc.	
190		Device control three	DC3	disc.		DC3	190	?	1310	disc.		disc.	

TABLE A-2/X.408

REFERENCE SET				CONVERTED SET									
Identification		Name or description ⓓ	Symbol	T.61		IA5IRV		ITA2		Facsimile		Videotex	
No.				Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.
200		Device control four	DC4	disc.		DC4	200	?	1310	disc.		disc.	
210		Negative acknowledgement	NAK	disc.		NAK	210	?	1310	disc.		disc.	
220		Synchronous idle	SYN	disc.		SYN	220	?	1310	disc.		disc.	
230		End of text block	ETB	disc.		ETB	230	?	1310	disc.		disc.	
240		Cancel	CAN	disc.		CAN	240	?	1310	disc.		CAN	240
250		End of medium	EM	disc.		EM	250	?	1310	disc.		disc.	
260	CM02	Substitute character	SUB	SUB	260	SUB	260	?	1310	FS		FS	
270	CE03	Escape	ESC	ESC	270	ESC	270	?	1310	FS		FS	
280		Information separator four	IS4	disc.		IS4	280	?	1310	disc.		disc.	
290		Information separator three	IS3	disc.		IS3	290	?	1310	disc.		disc.	
300		Information separator two	IS2	disc.		IS2	300	?	1310	disc.		disc.	
310		Information separator one	IS1	disc.		IS1	310	?	1310	disc.		disc.	
320													
330													
340													
350													
360													
370													
380													
390													

TABLE A-3/X.408

REFERENCE SET				CONVERTED SET									
Identification		Name or description ⓓ	Symbol	T.61		IA5IRV		ITA2		Facsimile		Videotex	
No.				Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.
400	CE07	Locking shift 0	LS0	LS0	400	FS		FS		FS		FS	
410	CE08	Locking shift 1	LS1	LS1	410	FS		FS		FS		FS	
420	CE09	Locking shift 1 right	LS1R	LS1R	420	FS		FS		FS		FS	
430	CE10	Locking shift 2	LS2	LS2	430	FS		FS		FS		FS	
440	CE11	Locking shift 2 right	LS2R	LS2R	440	FS		FS		FS		FS	
450	CE12	Locking shift 3	LS3	LS3	450	FS		FS		FS		FS	
460	CE13	Locking shift 3 right	LS3R	LS3R	460	FS		FS		FS		FS	
470													
480													
490													
500													
510													
520	CE04	Single shift 2	SS2	SS2	520	FS		FS		FS		FS	
530	CE05	Single shift 3	SS3	SS3	530	FS		FS		FS		FS	
540													
550													
560													
570													
580													
590													

TABLE A-4/X.408

REFERENCE SET				CONVERTED SET									
Identification		Name or description ⓓ	Symbol	T.61		IA5IRV		ITA2		Facsimile		Videotex	
No.				Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.
600		WHO ARE YOU? ⓐ	WRU	disc.		disc.		WRU	600	disc.		disc.	
610		National use ⓑ		disc.		disc.				disc.		disc.	
620		National use ⓑ		disc.		disc.				disc.		disc.	
630		National use ⓑ		disc.		disc.				disc.		disc.	
640													
650		Letter shift ⓒ	LS	disc.		disc.		LS	650	disc.		disc.	
660		Figure shift ⓒ	FS	disc.		disc.		FS	660	disc.		disc.	
670		All-space: null	NU	disc.		disc.		NU	670	disc.		disc.	
680													
690	CF16	Partial line down	PLD	PLD	690	disc.		disc.		disc.		disc.	
700	CF17	Partial line up	PLU	PLU	700	disc.		disc.		disc.		disc.	
710	CP06	Control sequence introducer	CSI	CSI	710	FS		disc.		disc.		FS	
720	CF20	Reverse line feed	RLF	RLF	720	disc.		disc.		disc.		disc.	
730	CP01	Page format selection	PFS	PFS	730	disc.		disc.		disc.		disc.	
740	CP03	Select graphic rendition	SGR	SGR	740	disc.		disc.		disc.		disc.	
750	CP04	Select horizontal spacing	SHS	SHS	750	disc.		disc.		disc.		disc.	
760	CP05	Select vertical spacing	SVS	SVS	760	disc.		disc.		disc.		disc.	
770	CP06	Select presentation direction	SPD	SPD	770	disc.		disc.		disc.		disc.	
780	CP07	Graphic size modification	GSM	GSM	780	disc.		disc.		disc.		disc.	
790	CM04	Identify graphic subrepertoire	IGS	IGS	790	disc.		disc.		disc.		disc.	

TABLE A-5/X.408

REFERENCE SET				CONVERTED SET									
Identification		Name or description ⓓ	Symbol	T.61		IA5IRV		ITA2		Facsimile		Videotex	
No.				Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.
800		Active position backward	APB	BS	80	BS	80	?	1310	FS		APB	800
810		Active position forward	APF	FS		FS		?	1310	FS		APF	810
820		Active position down	APD	LF	100	LF	100	LF	100	FS		APD	820
830		Active position up	APU	FS		FS		?	1310	FS		APU	830
840		Clear screen	CS	FF	120	FF	120	CR,LF	130,100	FS		CS	840
850		Active position return	APR	CR	130	CR	130	CR	130	FS		APR	850
860													
870													
880													
890													
900													
910													
920													
930													
940													
950													
960													
970													
980													
990													

TABLE A-6/X.408

REFERENCE SET				CONVERTED SET									
Identification		Name or description ⓓ	Symbol	T.61		IA5IRV		ITA2		Facsimile		Videotex	
No.				Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.
1000	SP01	Space	SP	SP	1000	SP	1000	SP	1000	SP	1000	SP	1000
1010	SP02	Exclamation mark	!	!	1010	!	1010	?	1310	!	1010	!	1010
1020	SP04	Quotation mark	"	"	1020	"	1020	?	1310	"	1020	"	1020
1030													
1040													
1050	SM02	Percent sign	%	%	1050	%	1050	?	1310	%	1050	%	1050
1060	SM03	Ampersand	&	&	1060	&	1060	?	1310	&	1060	&	1060
1070	SP05	Apostrophe	'	'	1070	'	1070	'	1070	'	1070	'	1070
1080	SP06	Left parenthesis	(	(	1080	(	1080	(	1080	(	1080	(	1080
1090	SP07	Right parenthesis	)	)	1090	)	1090	)	1090	)	1090	)	1090
1100	SM04	Asterisk	*	*	1100	*	1100	?	1310	*	1100	*	1100
1110	SA01	Plus sign	+	+	1110	+	1110	+	1110	+	1110	+	1110
1120	SP08	Comma	,	,	1120	,	1120	,	1120	,	1120	,	1120
1130	SP10	Hyphen or minus sign	–	–	1130	–	1130	–	1130	–	1130	–	1130
1140	SP11	Full stop, period	.	.	1140	.	1140	.	1140	.	1140	.	1140
1150	SP12	Solidus	/	/	1150	/	1150	/	1150	/	1150	/	1150
1160	ND10	Digit 0	0	0	1160	0	1160	0	1160	0	1160	0	1160
1170	ND01	Digit 1	1	1	1170	1	1170	1	1170	1	1170	1	1170
1180	ND02	Digit 2	2	2	1180	2	1180	2	1180	2	1180	2	1180
1190	ND03	Digit 3	3	3	1190	3	1190	3	1190	3	1190	3	1190

TABLE A-7/X.408

REFERENCE SET				CONVERTED SET									
Identification		Name or description	Symbol	T.61		IA5IRV		ITA2		Facsimile		Videotex	
No.				Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.
1200	ND04	Digit 4	4	4	1200	4	1200	4	1200	4	1200	4	1200
1210	ND05	Digit 5	5	5	1210	5	1210	5	1210	5	1210	5	1210
1220	ND06	Digit 6	6	6	1220	6	1220	6	1220	6	1220	6	1220
1230	ND07	Digit 7	7	7	1230	7	1230	7	1230	7	1230	7	1230
1240	ND08	Digit 8	8	8	1240	8	1240	8	1240	8	1240	8	1240
1250	ND09	Digit 9	9	9	1250	9	1250	9	1250	9	1250	9	1250
1260	SP13	Colon	:	:	1260	:	1260	:	1260	:	1260	:	1260
1270	SP14	Semicolon	;	;	1270	;	1270	?	1310	;	1270	;	1270
1280	SA03	Less-than sign	<	<	1280	<	1280	?	1310	<	1280	<	1280
1290	SA04	Equal sign	=	=	1290	=	1290	=	1290	=	1290	=	1290
1300	SA05	Grater-than sign	>	>	1300	>	1300	?	1310	>	1300	>	1300
1310	SP15	Question mark	?	?	1310	?	1310	?	1310	?	1310	?	1310
1320	SM05	Commercial at	@	@	1320	@	1320	?	1310	@	1320	@	1320
1330	LA02	Capital A	A	A	1330	A	1330	A	1330	A	1330	A	1330
1340	LB02	Capital B	B	B	1340	B	1340	B	1340	B	1340	B	1340
1350	LC02	Capital C	C	C	1350	C	1350	C	1350	C	1350	C	1350
1360	LD02	Capital D	D	D	1360	D	1360	D	1360	D	1360	D	1360
1370	LE02	Capital E	E	E	1370	E	1370	E	1370	E	1370	E	1370
1380	LF02	Capital F	F	F	1380	F	1380	F	1380	F	1380	F	1380
1390	LG02	Capital G	G	G	1390	G	1390	G	1390	G	1390	G	1390

TABLE A-8/X.408

REFERENCE SET				CONVERTED SET									
Identification		Name or description	Symbol	T.61		IA5IRV		ITA2		Facsimile		Videotex	
No.				Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.
1400	LH02	Capital H	H	H	1400	H	1400	H	1400	H	1400	H	1400
1410	LI02	Capital I	I	I	1410	I	1410	I	1410	I	1410	I	1410
1420	LJ02	Capital J	J	J	1420	J	1420	J	1420	J	1420	J	1420
1430	LK02	Capital K	K	K	1430	K	1430	K	1430	K	1430	K	1430
1440	LL02	Capital L	L	L	1440	L	1440	L	1440	L	1440	L	1440
1450	LM02	Capital M	M	M	1450	M	1450	M	1450	M	1450	M	1450
1460	LN02	Capital N	N	N	1460	N	1460	N	1460	N	1460	N	1460
1470	LO02	Capital O	O	O	1470	O	1470	O	1470	O	1470	O	1470
1480	LP02	Capital P	P	P	1480	P	1480	P	1480	P	1480	P	1480
1490	LQ02	Capital Q	Q	Q	1490	Q	1490	Q	1490	Q	1490	Q	1490
1500	LR02	Capital R	R	R	1500	R	1500	R	1500	R	1500	R	1500
1510	LS02	Capital S	S	S	1510	S	1510	S	1510	S	1510	S	1510
1520	LT02	Capital T	T	T	1520	T	1520	T	1520	T	1520	T	1520
1530	LU02	Capital U	U	U	1530	U	1530	U	1530	U	1530	U	1530
1540	LV02	Capital V	V	V	1540	V	1540	V	1540	V	1540	V	1540
1550	LW02	Capital W	W	W	1550	W	1550	W	1550	W	1550	W	1550
1560	LX02	Capital X	X	X	1560	X	1560	X	1560	X	1560	X	1560
1570	LY02	Capital Y	Y	Y	1570	Y	1570	Y	1570	Y	1570	Y	1570
1580	LZ02	Capital Z	Z	Z	1580	Z	1580	Z	1580	Z	1580	Z	1580
1590	SM06	Left square bracket	[	[	1590	[	1590	?	1310	[	1590	[	1590

TABLE A-9/X.408

REFERENCE SET				CONVERTED SET									
Identification		Name or description	Symbol	T.61		IA5IRV		ITA2		Facsimile		Videotex	
No.				Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.
1600		Reverse solidus	\	?	1310	\	1600	?	1310	\	1600	?	1310
1610	SM08	Right square bracket	]	]	1610	]	1610	?	1310	]	1610	]	1610
1620		Circumflex accent (IA5)	^	FS		^	1620	?	1310	^	1620	FS	
1630	SP09	Low line	_	_	1630	_	1630	?	1310	_	1630	_	1630
1640		Grave accent (IA5)	`	FS		`	1640	?	1310	`	1640	FS	
1650	LA01	Small a	a	a	1650	a	1650	A	1310	a	1650	a	1650
1660	LB01	Small b	b	b	1660	b	1660	B	1340	b	1660	b	1660
1670	LC01	Small c	c	c	1670	c	1670	C	1350	c	1670	c	1670
1680	LD01	Small d	d	d	1680	d	1680	D	1360	d	1680	d	1680
1690	LE01	Small e	e	e	1690	e	1690	E	1370	e	1690	e	1690
1700	LF01	Small f	f	f	1700	f	1700	F	1380	f	1700	f	1700
1710	LG01	Small g	g	g	1710	g	1710	G	1390	g	1710	g	1710
1720	LH01	Small h	h	h	1720	h	1720	H	1400	h	1720	h	1720
1730	LI01	Small i	i	i	1730	i	1730	I	1410	i	1730	i	1730
1740	LJ01	Small j	j	j	1740	j	1740	J	1420	j	1740	j	1740
1750	LK01	Small k	k	k	1750	k	1750	K	1430	k	1750	k	1750
1760	LL01	Small l	l	l	1760	l	1760	L	1440	l	1760	l	1760
1770	LM01	Small m	m	m	1770	m	1770	M	1450	m	1770	m	1770
1780	LN01	Small n	n	n	1780	n	1780	N	1460	n	1780	n	1780
1790	LO01	Small o	o	o	1790	o	1790	O	1470	o	1790	o	1790

TABLE A-10/X.408

REFERENCE SET				CONVERTED SET									
Identification		Name or description	Symbol	T.61		IA5IRV		ITA2		Facsimile		Videotex	
No.				Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.
1800	LP01	Small p	p	p	1800	p	1800	P	1480	p	1800	p	1800
1810	LQ01	Small q	q	q	1810	q	1810	Q	1490	q	1810	q	1810
1820	LR01	Small r	r	r	1820	r	1820	R	1500	r	1820	r	1820
1830	LS01	Small s	s	s	1830	s	1830	S	1510	s	1830	s	1830
1840	LT01	Small t	t	t	1840	t	1840	T	1520	t	1840	t	1840
1850	LU01	Small u	u	u	1850	u	1850	U	1530	u	1850	u	1850
1860	LV01	Small v	v	v	1860	v	1860	V	1540	v	1860	v	1860
1870	LW01	Small w	w	w	1870	w	1870	W	1550	w	1870	w	1870
1880	LX01	Small x	x	x	1880	x	1880	X	1560	x	1880	x	1880
1890	LY01	Small y	y	y	1890	y	1890	Y	1570	y	1890	y	1890
1900	LZ01	Small z	z	z	1900	z	1900	Z	1580	z	1900	z	1900
1910		Left curly bracket	{	<	1280	{	1910	?	1310	{	1910	<	1280
1920	SM13	Vertical line	|	|	1920	|	1920	?	1310	|	1920	|	1920
1930		Right curly bracket	}	>	1300	}	1930	?	1310	}	1930	>	1300
1940		Tilde, overline (IA5)	~	FS		~	1940	?	1310	~	1940	FS	
1950		Delete	DEL	disc.		DEL	1950	disc.		DEL	1950	disc.	
1960													
1970													
1980													
1990	—												

TABLE A-11/X.408

REFERENCE SET				CONVERTED SET									
Identification		Name or description	Symbol	T.61		IA5IRV		ITA2		Facsimile		Videotex	
No.				Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.
2000													
2010	SP03	Inverted exclamation mark	¡	¡	2010	?	1310	?	1310	¡	2010	¡	2010
2020	SC04	Cent sign	¢	¢	2020	¤	2080	?	1310	¢	2020	¢	2020
2030	SC02	Pound sign	£	£	2030	¤	2080	?	1310	£	2030	£	2030
2040	SC03	Dollar sign	$	$	2040	¤	2080	?	1310	$	2040	$	2040
2050	SC05	Yen sign	¥	¥	2050	¤	2080	?	1310	¥	2050	¥	2050
2060	SM01	Number sign	#	#	2060	#	2060	?	1310	#	2060	#	2060
2070	SM24	Section sign	§	§	2070	?	1310	?	1310	§	2070	§	2070
2080	SC01	Currency sign	¤	¤	2080	¤	2080	?	1310	¤	2080	¤	2080
2090	SP19	Single quotation mark left	‘	'	1070	'	1070	'	1070	'	1070	'	1070
2100	SP21	Double quotation mark left	“	"	1020	"	1020	?	1310	"	1020	"	1020
2110	SP17	Angle quotation mark left	« ‚	«	2110	<	1280	?	1310	«	2110	«	2110
2120	SM30	Leftward arrow	←	?	1310	?	1310	?	1310	?	1310	?	1310
2130	SM32	Upward arrow	↑	?	1310	?	1310	?	1310	?	1310	?	1310
2140	SM31	Rightward arrow	→	?	1310	?	1310	?	1310	?	1310	?	1310
2150	SM33	Downward arrow	↓	?	1310	?	1310	?	1310	?	1310	?	1310
2160	SM19	Degree sign	°	°	2160	?	1310	?	1310	°	2160	°	2160
2170	SA02	Plus/minus sign	±	±	2170	?	1310	?	1310	±	2170	±	2170
2180	NS02	Superscript 2	□²	□²	2180	?	1310	?	1310	□²	2180	□²	2180
2190	NS03	Superscript 3	□³	□³	2190	?	1310	?	1310	□³	2190	□³	2190

TABLE A-12/X.408

REFERENCE SET				CONVERTED SET									
Identification		Name or description	Symbol	T.61		IA5IRV		ITA2		Facsimile		Videotex	
No.				Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.
2200	SA07	Multiply sign	×	×	2200	?	1310	?	1310	×	2200	×	2200
2210	SM17	Micro sign	µ	µ	2210	?	1310	?	1310	µ	2210	µ	2210
2220	SM25	Paragraph sign, pilcrow	¶	¶	2220	?	1310	?	1310	¶	2220	¶	2220
2230	SM26	Middle dot	·	·	2230	?	1310	?	1310	·	2230	·	2230
2240	SA06	Divide sign	÷	÷	2240	?	1310	?	1310	÷	2240	÷	2240
2250	SP20	Single quotation mark right	’	’	1070	’	1070	’	1070	’	1070	’	1070
2260	SP22	Double quotation mark right	”	”	1020	”	1020	?	1310	”	1020	”	1020
2270	SP18	Angle quotation mark right	»	»	2270	>	1300	?	1310	»	2270	»	2270
2280	NF04	Fraction one quarter	¼	¼	2280	?	1310	?	1310	¼	2280	¼	2280
2290	NF01	Fraction one half	½	½	2290	?	1310	?	1310	½	2290	½	2290
2300	NF05	Fraction three quarter	¾	¾	2300	?	1310	?	1310	¾	2300	¾	2300
2310	SP16	Inverted questionmark left	¿	¿	2310	?	1310	?	1310	¿	2310	¿	2310
2320													
2330	SD13	Grave accent	`	`	2330	?	1310	?	1310	`	2330	`	2330
2340	SD11	Acute accent	´	´	2340	?	1310	?	1310	´	2340	´	2340
2350	SD15	Circumflex accent	^	^	2350	?	1310	?	1310	^	2350	^	2350
2360	SD19	Tilde	~	~	2360	?	1310	?	1310	~	2360	~	2360
2370	SD31	Macron	-	-	2370	?	1310	?	1310	-	2370	-	2370
2380	SD23	Breve	˘	˘	2380	?	1310	?	1310	˘	2380	˘	2380
2390	SD29	Dot	.	.	2390	?	1310	?	1310	.	2390	.	2390

TABLE A-13/X.408

REFERENCE SET				CONVERTED SET									
Identification		Name or description	Symbol	T.61		IA5IRV		ITA2		Facsimile		Videotex	
No.				Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.
2400	SD17	Diaeresis or umlaut mark	¨	¨	2400	?	1310	?	1310	¨	2400	¨	2400
2410													
2420	SD27	Ring	°	°	2420	?	1310	?	1310	°	2420	°	2420
2430	SD41	Cedilla	¸	¸	2430	?	1310	?	1310	¸	2430	¸	2430
2440													
2450	SD25	Double acute accent	˝	˝	2450	?	1310	?	1310	˝	2450	˝	2450
2460	SD43	Ogonek	˛	˛	2460	?	1310	?	1310	˛	2460	˛	2460
2470	SD21	Caron	ˇ	ˇ	2470	?	1310	?	1310	ˇ	2470	ˇ	2470
2480	SM12	Horizontal bar	–	?	1310	?	1310	?	1310	–	2480	–	2480
2490		Superscript 1	□[1]	?	1310	?	1310	?	1310	□[1]	2490	□[1]	2490
2500		Registered sign	®	?	1310	?	1310	?	1310	®	2500	®	2500
2510		Copyright sign	©	?	1310	?	1310	?	1310	©	2510	©	2510
2520		Trade mark sign	TM	?	1310	?	1310	?	1310	TM	2520	TM	2520
2530		Music note	♪	?	1310	?	1310	?	1310	♪	2530	♪	2530
2540													
2550													
2560													
2570													
2580													
2590													

TABLE A-14/X.408

REFERENCE SET				CONVERTED SET									
Identification		Name or description	Symbol	T.61		IA5IRV		ITA2		Facsimile		Videotex	
No.				Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.
2600		Fraction one eighth	1/8	?	1310	?	1310	?	1310	1/8	2600	1/8	2600
2610		Fraction three eighth	3/8	?	1310	?	1310	?	1310	3/8	2610	3/8	2610
2620		Fraction five eighth	5/8	?	1310	?	1310	?	1310	5/8	2620	5/8	2620
2630		Fraction seven eighth	7/8	?	1310	?	1310	?	1310	7/8	2630	7/8	2630
2640	SM18	Ohm sign	Ω	Ω	2640	?	1310	?	1310	Ω	2640	Ω	2640
2650	LA52	Capital Æ diphthong	Æ	Æ	2650	?	1310	?	1310	Æ	2650	Æ	2650
2660	LD62	Capital D with stroke	Đ	Đ	2660	?	1310	?	1310	Đ	2660	Đ	2660
2670	SM21	Ordinal indicator, feminine	a	a	2670	?	1310	?	1310	a	2670	a	2670
2680	LH62	Capital H with stroke	Ħ	Ħ	2680	?	1310	?	1310	Ħ	2680	Ħ	2680
2690													
2700	LI52	Capital IJ ligature	IJ	IJ	2700	?	1310	?	1310	IJ	2700	IJ	2700
2710	LL64	Capital L with middle dot	Ŀ	Ŀ	2710	?	1310	?	1310	Ŀ	2710	Ŀ	2710
2720	LL62	Capital L with stroke	Ł	Ł	2720	?	1310	?	1310	Ł	2720	Ł	2720
2730	LO62	Capital O with slash	Ø	Ø	2730	?	1310	?	1310	Ø	2730	Ø	2730
2740	LO52	Capital Œ ligature	Œ	Œ	2740	?	1310	?	1310	Œ	2740	Œ	2740
2750	SM20	Ordinal indicator, masculine	o	o	2750	?	1310	?	1310	o	2750	o	2750
2760	LT64	Capital thorn, icelandic	Þ	Þ	2760	?	1310	?	1310	Þ	2760	Þ	2760
2770	LT62	Capital T with stroke	Ŧ	Ŧ	2770	?	1310	?	1310	Ŧ	2770	Ŧ	2770
2780	LN62	Capital eng, lapp	Ŋ	Ŋ	2780	?	1310	?	1310	Ŋ	2780	Ŋ	2780
2790	LN63	Small n with apostrophe	ŉ	ŉ	2790	?	1310	?	1310	ŉ	2790	ŉ	2790

TABLE A-15/X.408

REFERENCE SET				CONVERTED SET									
Identification		Name or description	Symbol	T.61		IA5IRV		ITA2		Facsimile		Videotex	
No.				Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.
2800	LK61	Small k, greenlandic	ĸ	ĸ	2800	?	1310	?	1310	ĸ	2800	ĸ	2800
2810	LA51	Small æ diphthong	æ	æ	2810	?	1310	?	1310	æ	2810	æ	2810
2820	LD61	Small d with stroke	đ	đ	2820	?	1310	?	1310	đ	2820	đ	2820
2830	LD63	Small eth, icelandic	ð	ð	2830	?	1310	?	1310	ð	2830	ð	2830
2840	LH61	Small h with stroke	ħ	ħ	2840	?	1310	?	1310	ħ	2840	ħ	2840
2850	LI61	Small i without dot	i	i	2850	?	1310	?	1310	i	2850	i	2850
2860	LI51	Small ij ligature	ij	ij	2860	?	1310	?	1310	ij	2860	ij	2860
2870	LL63	Small l with middle dot	l·	l·	2870	?	1310	?	1310	l·	2870	l·	2870
2880	LL61	Small l with stroke	ł	ł	2880	?	1310	?	1310	ł	2880	ł	2880
2890	LO61	Small o with slash	ø	ø	2890	?	1310	?	1310	ø	2890	ø	2890
2900	LO51	Small œ ligature	œ	œ	2900	?	1310	?	1310	œ	2900	œ	2900
2910	LS61	Small sharp s, german	ß	ß	2910	?	1310	?	1310	ß	2910	ß	2910
2920	LT63	Small thorn, icelandic	þ	þ	2920	?	1310	?	1310	þ	2920	þ	2920
2930	LT61	Small t with stroke	ŧ	ŧ	2930	?	1310	?	1310	ŧ	2930	ŧ	2930
2940	LN61	Small eng, lapp	ŋ	ŋ	2940	?	1310	?	1310	ŋ	2940	ŋ	2940
2950													
2960													
2970													
2980													
2990													

TABLE A-16/X.408

REFERENCE SET				CONVERTED SET									
Identification		Name or description	Symbol	T.61		IA5IRV		ITA2		Facsimile		Videotex	
No.				Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.
3000	LA11	Small a with acute accent	á	á	3000	a	1650	A	1330	á	3000	á	3000
3010	LA12	Capital A with acute accent	Á	Á	3010	A	1330	A	1330	Á	3010	Á	3010
3020	LA13	Small a with grave accent	à	à	3020	a	1650	A	1330	à	3020	à	3020
3030	LA14	Capital A with grave accent	À	À	3030	A	1330	A	1330	À	3030	À	3030
3040	LA15	Small a with circumflex accent	â	â	3040	a	1650	A	1330	â	3040	â	3040
3050	LA16	Capital A with circumflex accent	Â	Â	3050	A	1330	A	1330	Â	3050	Â	3050
3060	LA17	Small a with diaeresis or umlaut mark	ä	ä	3060	a	1650	A	1330	ä	3060	ä	3060
3070	LA18	Capital A with diaeresis or umlaut mark	Ä	Ä	3070	A	1330	A	1330	Ä	3070	Ä	3070
3080	LA19	Small a with tilde	ã	ã	3080	a	1650	A	1330	ã	3080	ã	3080
3090	LA20	Capital A with tilde	Ã	Ã	3090	A	1330	A	1330	Ã	3090	Ã	3090
3100	LA23	Small a with breve	ă	ă	3100	a	1650	A	1330	ă	3100	ă	3100
3110	LA24	Capital A with breve	Ă	Ă	3110	A	1330	A	1330	Ă	3110	Ă	3110
3120	LA27	Small a with ring	å	å	3120	a	1650	A	1330	å	3120	å	3120
3130	LA28	Capital A with ring	Å	Å	3130	A	1330	A	1330	Å	3130	Å	3130
3140	LA31	Small a with macron	ā	ā	3140	a	1650	A	1330	ā	3140	ā	3140
3150	LA32	Capital A with macron	Ā	Ā	3150	A	1330	A	1330	Ā	3150	Ā	3150
3160	LA43	Small a with ogonek	ą	ą	3160	a	1650	A	1330	ą	3160	ą	3160
3170	LA44	Capital A with ogonek	Ą	Ą	3170	A	1330	A	1330	Ą	3170	Ą	3170
3180	LC11	Small c with acute accent	ć	ć	3180	c	1670	C	1350	ć	3180	ć	3180
3190	LC12	Capital C with acute accent	Ć	Ć	3190	C	1350	C	1350	Ć	3190	Ć	3190

TABLE A-17/X.408

REFERENCE SET				CONVERTED SET									
Identification		Name or description	Symbol	T.61		IA5IRV		ITA2		Facsimile		Videotex	
No.				Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.
3200	LC15	Small c with circumflex accent	ĉ	ĉ	3200	c	1670	C	1350	ĉ	3200	ĉ	3200
3210	LC16	Capital C with circumflex accent	Ĉ	Ĉ	3210	C	1350	C	1350	Ĉ	3210	Ĉ	3210
3220	LC21	Small c with caron	č	č	3220	c	1670	C	1350	č	3220	č	3220
3230	LC22	Capital C with caron	Č	Č	3230	C	1350	C	1350	Č	3230	Č	3230
3240	LC29	Small c with dot	ċ	ċ	3240	c	1670	C	1350	ċ	3240	ċ	3240
3250	LC30	Capital C with dot	Ċ	Ċ	3250	C	1350	C	1350	Ċ	3250	Ċ	3250
3260	LC41	Small c with cedilla	ç	ç	3260	c	1670	C	1350	ç	3260	ç	3260
3270	LC42	Capital C with cedilla	Ç	Ç	3270	C	1350	C	1350	Ç	3270	Ç	3270
3280	LD21	Small d with caron	ď	ď	3280	d	1680	D	1360	ď	3280	ď	3280
3290	LD22	Capital D with caron	Ď	Ď	3290	D	1360	D	1360	Ď	3290	Ď	3290
3300	LE11	Small e with acute accent	é	é	3300	e	1690	E	1370	é	3300	é	3300
3310	LE12	Capital E with acute accent	É	É	3310	E	1370	E	1370	É	3310	É	3310
3320	LE13	Small e with grave accent	è	è	3320	e	1690	E	1370	è	3320	è	3320
3330	LE14	Capital E with grave accent	È	È	3330	E	1370	E	1370	È	3330	È	3330
3340	LE15	Small e with circumflex accent	ê	ê	3340	e	1690	E	1370	ê	3340	ê	3340
3350	LE16	Capital E with circumflex accent	Ê	Ê	3350	E	1370	E	1370	Ê	3350	Ê	3350
3360	LE17	Small e with diaeresis or umlaut mark	ë	ë	3360	e	1690	E	1370	ë	3360	ë	3360
3370	LE18	Capital E with diaeresis or umlaut mark	Ë	Ë	3370	E	1370	E	1370	Ë	3370	Ë	3370
3380	LE21	Small e with caron	ě	ě	3380	e	1690	E	1370	ě	3380	ě	3380
3390	LE22	Capital E with caron	Ě	Ě	3390	E	1370	E	1370	Ě	3390	Ě	3390

TABLE A-18/X.408

REFERENCE SET				CONVERTED SET									
Identification		Name or description	Symbol	T.61		IA5IRV		ITA2		Facsimile		Videotex	
No.				Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.
3400	LE29	Small e with dot	ė	ė	3400	e	1690	E	1370	ė	3400	ė	3400
3410	LE30	Capital E with dot	Ė	Ė	3410	E	1370	E	1370	Ė	3410	Ė	3410
3420	LE31	Small e with macron	ē	ē	3420	e	1690	E	1370	ē	3420	ē	3420
3430	LE32	Capital E with macron	Ē	Ē	3430	E	1370	E	1370	Ē	3430	Ē	3430
3440	LE43	Small e with ogonek	ę	ę	3440	e	1690	E	1370	ę	3440	ę	3440
3450	LE44	Capital E with ogonek	Ę	Ę	3450	E	1370	E	1370	Ę	3450	Ę	3450
3460	LG11	Small g with acute accent	ǵ	ǵ	3460	g	1710	G	1390	ǵ	3460	ǵ	3460
3470	LG15	Small g with circumflex accent	ĝ	ĝ	3470	g	1710	G	1390	ĝ	3470	ĝ	3470
3480	LG16	Capital G with circumflex accent	Ĝ	Ĝ	3480	G	1390	G	1390	Ĝ	3480	Ĝ	3480
3490	LG23	Small g with breve	ğ	ğ	3490	g	1710	G	1390	ğ	3490	ğ	3490
3500	LG24	Capital G with breve	Ğ	Ğ	3500	G	1390	G	1390	Ğ	3500	Ğ	3500
3510	LG29	Small g with dot	ġ	ġ	3510	g	1710	G	1390	ġ	3510	ġ	3510
3520	LG30	Capital G with dot	Ġ	Ġ	3520	G	1390	G	1390	Ġ	3520	Ġ	3520
3530	LG42	Capital G with cedilla	Ģ	Ģ	3530	G	1390	G	1390	Ģ	3530	Ģ	3530
3540	LH15	Small h with circumflex accent	ĥ	ĥ	3540	h	1720	H	1400	ĥ	3540	ĥ	3540
3550	LH16	Capital H with circumflex accent	Ĥ	Ĥ	3550	H	1400	H	1400	Ĥ	3550	Ĥ	3550
3560	LI11	Small i with acute accent	í	í	3560	i	1730	I	1410	í	3560	í	3560
3570	LI12	Capital I with acute accent	Í	Í	3570	I	1410	I	1410	Í	3570	Í	3570
3580	LI13	Small i with grave accent	ì	ì	3580	i	1730	I	1410	ì	3580	ì	3580
3590	LI14	Capital I with grave accent	Ì	Ì	3590	I	1410	I	1410	Ì	3590	Ì	3590

TABLE A-19/X.408

REFERENCE SET				CONVERTED SET									
Identification		Name or description	Symbol	T.61		IA5IRV		ITA2		Facsimile		Videotex	
No.				Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.
3600	LI15	Small i with circumflex accent	î	î	3600	i	1730	I	1410	î	3600	î	3600
3610	LI16	Capital I with circumflex accent	Î	Î	3610	I	1410	I	1410	Î	3610	Î	3610
3620	LI17	Small i with diaeresis or umlaut mark	ï	ï	3620	i	1730	I	1410	ï	3620	ï	3620
3630	LI18	Capital I with diaeresis or umlaut mark	Ï	Ï	3630	I	1410	I	1410	Ï	3630	Ï	3630
3640	LI19	Small i with tilde	ĩ	ĩ	3640	i	1730	I	1410	ĩ	3640	ĩ	3640
3650	LI20	Capital I with tilde	Ĩ	Ĩ	3650	I	1410	I	1410	Ĩ	3650	Ĩ	3650
3660	LI30	Capital I with dot	İ	İ	3660	I	1410	I	1410	İ	3660	İ	3660
3670	LI31	Small i with macron	ī	ī	3670	i	1730	I	1410	ī	3670	ī	3670
3680	LI32	Capital I with macron	Ī	Ī	3680	I	1410	I	1410	Ī	3680	Ī	3680
3690	LI43	Small i with ogonek	į	į	3690	i	1730	I	1410	į	3690	į	3690
3700	LI44	Capital I with ogonek	Į	Į	3700	I	1410	I	1410	Į	3700	Į	3700
3710	LJ15	Small j with circumflex accent	ĵ	ĵ	3710	j	1740	J	1420	ĵ	3710	ĵ	3710
3720	LJ16	Capital J with circumflex accent	Ĵ	Ĵ	3720	J	1420	J	1420	Ĵ	3720	Ĵ	3720
3730	LK41	Small k with cedilla	ķ	ķ	3730	k	1750	K	1430	ķ	3730	ķ	3730
3740	LK42	Capital K with cedilla	Ķ	Ķ	3740	K	1430	K	1430	Ķ	3740	Ķ	3740
3750	LL11	Small l with acute accent	ĺ	ĺ	3750	l	1760	L	1440	ĺ	3750	ĺ	3750
3760	LL12	Capital L acute accent	Ĺ	Ĺ	3760	L	1440	L	1440	Ĺ	3760	Ĺ	3760
3770	LL21	Small l with caron	ľ	ľ	3770	l	1760	L	1440	ľ	3770	ľ	3770
3780	LL22	Capital L with caron	Ľ	Ľ	3780	L	1440	L	1440	Ľ	3780	Ľ	3780
3790	LN11	Small n with acute accent	ń	ń	3790	n	1780	N	1460	ń	3790	ń	3790

TABLE A-20/X.408

REFERENCE SET				CONVERTED SET									
Identification		Name or description	Symbol	T.61		IA5IRV		ITA2		Facsimile		Videotex	
No.				Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.
3800	LN12	Capital N with acute accent	Ń	Ń	3800	N	1460	N	1460	Ń	3800	Ń	3800
3810	LN19	Small n with tilde	ñ	ñ	3810	n	1780	N	1460	ñ	3810	ñ	3810
3820	LN20	Capital N with tilde	Ñ	Ñ	3820	N	1460	N	1460	Ñ	3820	Ñ	3820
3830	LN21	Small n with caron	ň	ň	3830	n	1780	N	1460	ň	3830	ň	3830
3840	LN22	Capital N with caron	Ň	Ň	3840	N	1460	N	1460	Ň	3840	Ň	3840
3850	LN41	Small n with cedilla	ņ	ņ	3850	n	1780	N	1460	ņ	3850	ņ	3850
3860	LN42	Capital N with cedilla	Ņ	Ņ	3860	N	1460	N	1460	Ņ	3860	Ņ	3860
3870	LO11	Small o with acute accent	ó	ó	3870	o	1790	O	1470	ó	3870	ó	3870
3880	LO12	Capital O with acute accent	Ó	Ó	3880	O	1470	O	1470	Ó	3880	Ó	3880
3890	LO13	Small o with grave accent	ò	ò	3890	o	1790	O	1470	ò	3890	ò	3890
3900	LO14	Capital O with grave accent	Ò	Ò	3900	O	1470	O	1470	Ò	3900	Ò	3900
3910	LO15	Small o with circumflex accent	ô	ô	3910	o	1790	O	1470	ô	3910	ô	3910
3920	LO16	Capital O with circumflex accent	Ô	Ô	3920	O	1470	O	1470	Ô	3920	Ô	3920
3930	LO17	Small o with diaeresis or umlaut mark	ö	ö	3930	o	1790	O	1470	ö	3930	ö	3930
3940	LO18	Capital O with diaeresis or umlaut mark	Ö	Ö	3940	O	1470	O	1470	Ö	3940	Ö	3940
3950	LO19	Small o with tilde	õ	õ	3950	o	1790	O	1470	õ	3950	õ	3950
3960	LO20	Capital O with tilde	Õ	Õ	3960	O	1470	O	1470	Õ	3960	Õ	3960
3970	LO25	Small o with double acute accent	ő	ő	3970	o	1790	O	1470	ő	3970	ő	3970
3980	LO26	Capital O with double acute accent	Ő	Ő	3980	O	1470	O	1470	Ő	3980	Ő	3980
3990	LO31	Small o with macron	ō	ō	3990	o	1790	O	1470	ō	3990	ō	3990

TABLE A-21/X.408

REFERENCE SET				CONVERTED SET									
Identification		Name or description	Symbol	T.61		IA5IRV		ITA2		Facsimile		Videotex	
No.				Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.
4000	L032	Capital O with macron	Ō	Ō	4000	O	1470	O	1470	Ō	4000	Ō	4000
4010	LR11	Small r with acute accent	ŕ	ŕ	4010	r	1820	R	1500	ŕ	4010	ŕ	4010
4020	LR12	Capital R with acute accent	Ŕ	Ŕ	4020	R	1500	R	1500	Ŕ	4020	Ŕ	4020
4030	LR21	Small r with caron	ř	ř	4030	r	1820	R	1500	ř	4030	ř	4030
4040	LR22	Capital R with caron	Ř	Ř	4040	R	1500	R	1500	Ř	4040	Ř	4040
4050	LR41	Small r with cedilla	ŗ	ŗ	4050	r	1820	R	1500	ŗ	4050	ŗ	4050
4060	LR42	Capital R with cedilla	Ŗ	Ŗ	4060	R	1500	R	1500	Ŗ	4060	Ŗ	4060
4070	LS11	Small s with acute accent	ś	ś	4070	s	1830	S	1510	ś	4070	ś	4070
4080	LS12	Capital S with acute accent	Ś	Ś	4080	S	1510	S	1510	Ś	4080	Ś	4080
4090	LS15	Small s with circumflex accent	ŝ	ŝ	4090	s	1830	S	1510	ŝ	4090	ŝ	4090
4100	LS16	Capital S with circumflex accent	Ŝ	Ŝ	4100	S	1510	S	1510	Ŝ	4100	Ŝ	4100
4110	LS21	Small s with caron	š	š	4110	s	1830	S	1510	š	4110	š	4110
4120	LS22	Capital S with caron	Š	Š	4120	S	1510	S	1510	Š	4120	Š	4120
4130	LS41	Small s with cedilla	ş	ş	4130	s	1830	S	1510	ş	4130	ş	4130
4140	LS42	Capital S with cedilla	Ş	Ş	4140	S	1510	S	1510	Ş	4140	Ş	4140
4150	LT21	Small t with caron	ť	ť	4150	t	1840	T	1520	ť	4150	ť	4150
4160	LT22	Capital T with caron	Ť	Ť	4160	T	1520	T	1520	Ť	4160	Ť	4160
4170	LT41	Small t with cedilla	ţ	ţ	4170	t	1840	T	1520	ţ	4170	ţ	4170
4180	LT42	Capital T with cedilla	Ţ	Ţ	4180	T	1520	T	1520	Ţ	4180	Ţ	4180
4190	LU11	Small u with acute accent	ú	ú	4190	u	1850	U	1530	ú	4190	ú	4190

TABLE A-22/X.408

REFERENCE SET				CONVERTED SET									
Identification		Name or description	Symbol	T.61		IA5IRV		ITA2		Facsimile		Videotex	
No.				Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.
4200	LU12	Capital U with acute accent	Ú	Ú	4200	U	1530	U	1530	Ú	4200	Ú	4200
4210	LU13	Small u with grave accent	ù	ù	4210	u	1850	U	1530	ù	4210	ù	4210
4220	LU14	Capital U with grave accent	Ù	Ù	4220	U	1530	U	1530	Ù	4220	Ù	4220
4230	LU15	Small u with circumflex accent	û	û	4230	u	1850	U	1530	û	4230	û	4230
4240	LU16	Capital U with circumflex accent	Û	Û	4240	U	1530	U	1530	Û	4240	Û	4240
4250	LU17	Small u with diaeresis or umlaut mark	ü	ü	4250	u	1850	U	1530	ü	4250	ü	4250
4260	LU18	Capital U with diaeresis or umlaut mark	Ü	Ü	4260	U	1530	U	1530	Ü	4260	Ü	4260
4270	LU19	Small u with tilde	ũ	ũ	4270	u	1850	U	1530	ũ	4270	ũ	4270
4280	LU20	Capital U with tilde	Ũ	Ũ	4280	U	1530	U	1530	Ũ	4280	Ũ	4280
4290	LU23	Small u with breve	ŭ	ŭ	4290	u	1850	U	1530	ŭ	4290	ŭ	4290
4300	LU24	Capital U with breve	Ŭ	Ŭ	4300	U	1530	U	1530	Ŭ	4300	Ŭ	4300
4310	LU25	Small u with double acute accent	ű	ű	4310	u	1850	U	1530	ű	4310	ű	4310
4320	LU26	Capital U with double acute accent	Ű	Ű	4320	U	1530	U	1530	Ű	4320	Ű	4320
4330	LU27	Small u with ring	ů	ů	4330	u	1850	U	1530	ů	4330	ů	4330
4340	LU28	Capital U with ring	Ů	Ů	4340	U	1530	U	1530	Ů	4340	Ů	4340
4350	LU31	Small u with macron	ū	ū	4350	u	1850	U	1530	ū	4350	ū	4350
4360	LU32	Capital U with macron	Ū	Ū	4360	U	1530	U	1530	Ū	4360	Ū	4360
4370	LU43	Small u with ogonek	ų	ų	4370	u	1850	U	1530	ų	4370	ų	4370
4380	LU44	Capital U with ogonek	Ų	Ų	4380	U	1530	U	1530	Ų	4380	Ų	4380
4390	LW15	Small w with circumflex accent	ŵ	ŵ	4390	w	1870	W	1550	ŵ	4390	ŵ	4390

TABLE A-23/X.408

REFERENCE SET				CONVERTED SET									
Identification		Name or description	Symbol	T.61		IA5IRV		ITA2		Facsimile		Videotex	
No.				Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.	Symbol	No.
4400	LW16	Capital W with circumflex accent	Ŵ	Ŵ	4400	W	1550	W	1550	Ŵ	4400	Ŵ	4400
4410	LY11	Small y with acute accent	ý	ý	4410	y	1890	Y	1570	ý	4410	ý	4410
4420	LY12	Capital Y with acute accent	Ý	Ý	4420	Y	1570	Y	1570	Ý	4420	Ý	4420
4430	LY15	Small y with circumflex accent	ŷ	ŷ	4430	y	1890	Y	1570	ŷ	4430	ŷ	4430
4440	LY16	Capital Y with circumflex accent	Ŷ	Ŷ	4440	Y	1570	Y	1570	Ŷ	4440	Ŷ	4440
4450	LY17	Small y with diaeresis or umlaut mark	ÿ	ÿ	4450	y	1890	Y	1570	ÿ	4450	ÿ	4450
4460	LY18	Capital Y with diaeresis or umlaut mark	Ÿ	Ÿ	4460	Y	1570	Y	1570	Ÿ	4460	Ÿ	4460
4470	LZ11	Small z with acute accent	ź	ź	4470	z	1900	Z	1580	ź	4470	ź	4470
4480	LZ12	Capital Z with acute accent	Ź	Ź	4480	Z	1580	Z	1580	Ź	4480	Ź	4480
4490	LZ21	Small z with caron	ž	ž	4490	z	1900	Z	1580	ž	4490	ž	4490
4500	LZ22	Capital Z with caron	Ž	Ž	4500	Z	1580	Z	1580	Ž	4500	Ž	4500
4510	LZ29	Small z with dot	ż	ż	4510	z	1900	Z	1580	ż	4510	ż	4510
4520	LZ30	Capital Z with dot	Ż	Ż	4520	Z	1580	Z	1580	Ż	4520	Ż	4520
4530													
4540													
4550													
4560													
4570													
4580													
4590													

ANNEX B

(to Recommendation X.408)

Abbreviations

The following abbreviations are used in this Recommendation.

APD	Active Position Down
APR	Active Position Return
BS	Backspace
CR	Carriage Return
CS	Clear Screen
FF	Form Feed
FS	Further Study
G3	Group 3
G3Fax	Group 3 Facsimile Type
G4	Group 4
HT	Horizontal Tabulation
IA	International Alphabet
IGS	Identify Graphic Subrepertoire
ITA	International Telegraph Alphabet
LF	Line Feed
MHS	Message Handling System
OSI	Open Systems Interconnection
PFS	Page Format Selection
PLD	Partial Line Down
PLU	Partial Line Up
RLF	Reverse Line Feed
SGR	Select Graphic Rendition
SHS	Select Horizontal Spacing
SP	Space
SUB	Substitute Character
SVS	Select Vertical Spacing
TLX	Telex Type
TTX	Teletex Type
VT	Vertical Tabulation

Recommendation X.411

MESSAGE HANDLING SYSTEMS: MESSAGE TRANSFER SYSTEM: ABSTRACT SERVICE DEFINITION AND PROCEDURES[1]

(Malaga-Torremolinos, 1984; amended at Melbourne, 1988)

The establishment in various countries of telematic services and computer-based store-and-forward message services in association with public data networks creates a need to produce standards to facilitate international message exchange between subscribers to such services.

The CCITT,

considering

(a) the need for Message Handling systems;

(b) the need to transfer messages of different types;

(c) that Recommendation X.200 defines the reference model of open systems interconnection for CCITT applications;

(d) that Recommendations X.208, X.217, X.218 and X.219 provide the foundation for CCITT applications;

(e) that the X.500-series Recommendations define directory systems;

(f) that Message Handling systems are defined in a series of Recommendations: X.400, X.402, X.403, X.407, X.408, X.411, X.413 and X.419; and

(g) that interpersonal messaging is defined in RecommendationsX.420 and T.330,

unanimously declares

(1) that the message transfer system (MTS) abstract service is defined in Section 2;

(2) that the message transfer agent (MTA) abstract service is defined in Section 3;

(3) that the procedures performed by message-transfer-agents (MTAs) to ensure that correct distributed operation of the message transfer system (MTS) are defined in Section 4.

TABLE OF CONTENTS

[1] Recommendation X.411 and ISO 10021-4, Information Processing Systems – Text Communication – MOTIS – Message Transfer System: Abstract Service Definition and Procedures, were developed in close collaboration and are technically aligned, except for the differences noted in Annex C.

SECTION 1 – INTRODUCTION

0 Introduction

This Recommendation is one of a set of Recommendations defining Message Handling in a distributed open systems environment.

Message Handling provides for the exchange of messages between users on a store-and-forward basis. A message submitted by one user (the *originator*) is transferred through the message transfer system (MTS) and delivered to one or more other users (the *recipients*).

The MTS comprises a number of message-transfer-agents (MTAs), which transfer messages and deliver them to their intended recipients.

This Recommendation was developed jointly by CCITT and ISO. The equivalent ISO document is ISO 10021-4.

1 Scope

This Recommendation defines the abstract service provided by the MTS (the MTS abstract service), and specifies the procedures to be performed by MTAs to ensure the correct distributed operation of the MTS.

Recommendation X.402 identifies the other Recommendations which define other aspects of Message Handling Systems.

Access to the MTS abstract service defined in this Recommendation may be provided by the MTS access protocol (P3) defined in Recommendation X.419. The distributed operation of the MTS defined in this Recommendation may be provided by the use of the MTS transfer protocol (P1) also defined in Recommendation X.419.

Section 2 of this Recommendation defines the MTS abstract service. Paragraph 6 describes the message transfer system model. Paragraph 7 provides an overview of the MTS abstract service. Paragraph 8 defines the semantics of the parameters of the MTS abstract service. Paragraph 9 defines the abstract syntax of the MTA abstract service.

Section 3 of this Recommendation defines the MTA abstract service. Paragraph 10 refines the model of the MTS, first presented in § 6, to show that the MTS comprises a number of MTAs that interwork with one another to provide the MTS abstract service. Paragraph 11 provides an overview of the MTA abstract service. Paragraph 12 defines the semantics of the parameters of the MTA abstract service. Paragraph 13 defines the abstract-syntax of the MTA abstract service.

Section 4 of this Recommendation specifies the procedures performed by MTAs to ensure the correct distributed operation of the MTS.

Annex A provides a reference definition of the MTS object identifiers cited in the ASN.1 modules of this Recommendation.

Annex B provides a reference definition of the upper bounds of the size constraints imposed upon variable length data types defined in ASN.1 modules in the body of this Recommendation.

Annex C identifies the technical differences between ISO/IEC and CCITT versions of CCITT Recommendations X.411 and ISO/IEC 10021-4.

2 References

References are listed in Recommendation X.402.

3 Definitions

Definitions are listed in Recommendation X.402.

4 Abbreviations

Abbreviations are listed in Recommendation X.402.

5 Conventions

This Recommendation uses the descriptive conventions described below.

5.1 *Terms*

Throughout this Recommendation, the words of defined terms and the names and values of the parameters of the MTS abstract service and the MTA abstract service, unless they are proper names, begin with a lower-case letter and are linked by a hyphen thus: defined-term. Proper names begin with an upper-case letter and are not linked by a hyphen thus: Proper Name. In §§ 8 and 12, the names and values of the parameters of the MTS abstract service and the MTA abstract service are printed in bold.

5.2 *Presence of parameters*

In the Tables of parameters in §§ 8 and 12, the presence of each parameter is qualified as follows:

- Mandatory (M): A mandatory parameter shall always be present.
- Optional (O): An optional argument shall be present at the direction of the invoker of the abstract-operation; an optional result at the discretion of the performer of the abstract-operation.
- Conditional (C): A conditional parameter shall be present as defined by this [Recommendation/International Standard].

Where a conditional parameter shall be present due to some action on the message, probe or report by the MTS, this is explicitly defined. The presence of other conditional parameters is dependent on the presence of those parameters in other abstract-operations (for example, the presence of a conditional argument of the Message-transfer abstract-operation is dependent on the presence of the same optional argument in the related Message-submission abstract-operation).

5.3 *Abstract syntax definitions*

This Recommendation defines the abstract-syntax of the MTS abstract service and the MTA abstract service using the abstract syntax notation (ASN.1) defined in Recommendation X.208, and the abstract service definition conventions defined in Recommendation X.407.

Where there are changes implied to the protocols defined in Recommendation X.411 (1984), these are highlighted in the abstract syntax definitions by means of underlining.

6 Message transfer system model

Message Handling provides for the exchange of messages between users on a store-and-forward basis. A message submitted by one user (the *originator*) is transferred through the message transfer system (MTS) and delivered to one or more other users (the *recipients*).

The MTS is described using an abstract model in order to define the services provided by the MTS as a whole – the MTS abstract service.

The MTS is modelled as an *object*, whose overall behaviour can be described without reference to its internal structure. The services provided by the MTS object are made available at *ports*. A type of port represents a particular view of the services provided by the MTS object.

A user of the MTS is also modelled as an object, which obtains the services provided by the MTS through a port which is *paired* with an MTS port of the same type.

A type of port corresponds to a set of a *abstract-operations* which can occur at the port; those which can be performed by the MTS object (invoked by the MTS-user object), and those which can be invoked by the MTS object (performed by the MTS-user object).

A port may be symmetrical, in which case the set of operations performed by the MTS object may also be invoked by the MTS object, and vice versa. Otherwise, the port is asymmetrical, in which case the object is said to be the *supplier* or *consumer* with respect to the type of port. The terms *supplier* and *consumer* are used only to distinguish between the roles of a pair of ports in invoking or performing operations. The assignment of the terms is usually intuitive when one object is providing a service used by another object; the service object (e.g., the MTS) is usually regarded as being the *supplier*, and the user object (e.g., an MTS-user object) is usually regarded as being the *consumer*.

Before objects can invoke operations on one another, they must be bound into an abstract *association*. The binding of an assocation between the objects establishes a relationship between the objects which lasts until the association is released. An association is always released by the initiator of the association. The binding of an association establishes the *credentials* of the objects to interact, and the *application-context* and *security-context* of the association. The *application-context* of an association may be one or more types of port paired between two objects.

The model presented is abstract. That is, it is not always possible for an outside observer to identify the boundaries between objects, or to decide on the moment or the means by which operations occur. However, in some cases the abstract model will be *realised*. For example, a pair of objects which communicate through paired ports may be located in different open systems. In this case, the boundary between the objects is visible, the ports are exposed, and the operations may be supported by instances of OSI communication.

The MTS object supports ports of three different types: a *submission-port*, a *delivery-port* and an *administration-port*.

A submission-port enables an MTS-user to submit messages to the MTS for transfer and delivery to one or more recipient MTS-users, and to probe the ability of the MTS to deliver a subject-message.

A delivery-port enables an MTS-user to accept delivery of messages from the MTS, and to accept reports on the delivery or non-delivery of messages and probes.

An administration-port enables an MTS-user to change long term parameters held by the MTS associated with message delivery, and enables either the MTS or the MTS-user to change their *credentials* with one another.

A message submitted by one MTS-user via a submission-port will normally be delivered to one or more recipient MTS-users via delivery ports. The originating MTS-user may elect to be notified of the delivery of a message via its delivery-port.

Figure 1/X.411 models the message transfer system (MTS).

Paragraph 7 provides an overview of the MTS Abstract Service.

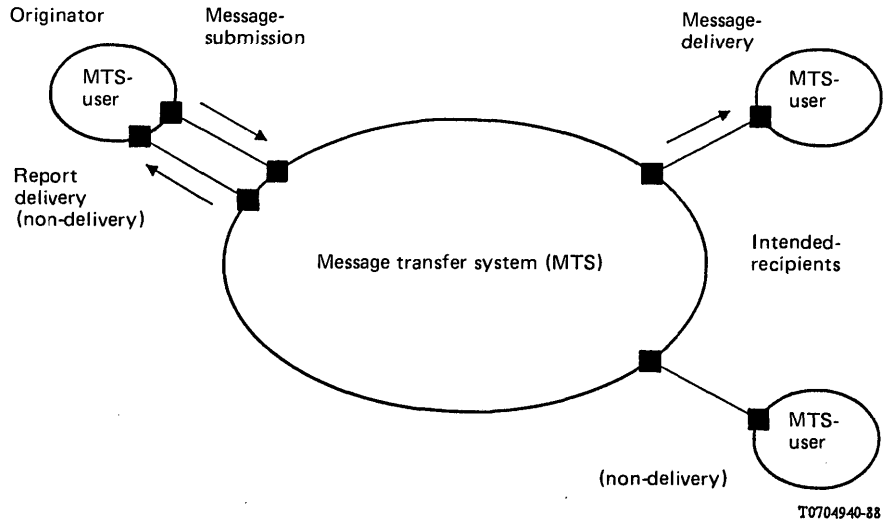


FIGURE 1/X.411

Message transfer system model

7 Message transfer system abstract service overview

This Recommendation defines the following services that comprise the MTS abstract service:

MTS bind and unbind

a) MTS-bind

b) MTS-unbind

Submission port abstract operations

c) Message-submission

d) Probe-submission

e) Cancel-deferred-delivery

f) Submission-control

Delivery port abstract operations

g) Message-delivery

h) Report-delivery

i) Delivery-control

Administration port abstract operations

j) Register

k) Change-credentials.

7.1 *MTS bind and unbind*

The **MTS-bind** enables either the MTS-user to establish an association with the MTS, or the MTS to establish an association with the MTS-user. Abstract-operations other than MTS-bind can only be invoked in the context of an established association.

The **MTS-unbind** enables the release of an established association by the initiator of the association.

7.2 *Submission port*

The **message-submission** abstract-operation enables an MTS-user to submit a message to the MTS for transfer and delivery to one or more recipient MTS-users.

The **probe-submission** abstract-operation enables an MTS-user to submit a probe in order to determine whether or not a message could be transferred and delivered to one or more recipient MTS-users if it were to be submitted.

The **cancel-deferred-delivery** abstract-operation enables an MTS-user to request cancellation of a message previously submitted (for deferred delivery) by invocation of the message-submission-abstract-operation.

The **submission-control** abstract-operation enables the MTS to constrain the use of the submission-port abstract-operations by the MTS-user.

The **message-submission** and **Probe-submission** abstract-operations may cause subsequent invocation of the Report-delivery abstract-operation by the MTS.

7.3 *Delivery port*

The **message-delivery** abstract-operation enables the MTS to deliver a message to the MTS-user.

The **report-delivery** abstract-operation enables the MTS to acknowledge to the MTS-user the outcome of a previous invocation of the message-submission or probe-submission abstract-operations. For the message-submission abstract-operation, the report-delivery abstract-operation indicates the delivery or non-delivery of the submitted message. For the probe-submission abstract-operation, the report-delivery abstract-operation indicates whether or not a message could be delivered if it were to be submitted. The report-delivery abstract-operation may also convey a notification of physical-delivery by a PDS.

The **delivery-control** abstract-operation enables an MTS-user to constrain the use of the delivery-port abstract-operations by the MTS.

7.4 *Administration port*

The **register** abstract-operation enables an MTS-user to change long term parameters of the MTS-user held by the MTS, associated with message delivery.

The **change-credentials** abstract-operation enables either an MTS-user to change its **credentials** with the MTS, or the MTS to change its **credentials** with the MTS-user.

8 Message transfer system abstract service definition

This section defines the semantics of the parameters of the MTS abstract service.

Paragraph 8.1 defines the MTS-bind and MTS-unbind. Paragraph 8.2 defines the submission-port. Paragraph 8.3 defines the delivery-port. Paragraph 8.4 defines the administration-port. Paragraph 8.5 defines some common parameter types.

The abstract-syntax of the MTS abstract service is defined in § 9.

8.1 *MTS-bind and MTS-unbind*

This section defines the MTS-bind and MTS-unbind used to establish and release associations between an MTS-user and the MTS.

8.1.1 *Abstract-bind and abstract-unbind*

This section defines the following abstract-bind and abstract-unbind operations:

a) MTS-bind

b) MTS-unbind.

8.1.1.1 *MTS-bind*

The MTS-bind enables an MTS-user to establish an association with the MTS, or the MTS to establish an association with an MTS-user.

The MTS-bind establishes the **credentials** of an MTS-user and the MTS to interact, and the **application-context** and **security-context** of the association. An association can only be released by the initiator of that association (using MTS-unbind).

Abstract-operations other than MTS-bind can only be invoked in the context of an established association.

The successful completion of the MTS-bind signifies the establishment of an association.

The disruption of the MTS-bind by a bind-error indicates that an association has not been established.

8.1.1.1.1 *Arguments*

Table 1/X.411 lists the arguments of the MTS-bind, and for each argument qualifies its presence and indicates the clause in which the argument is defined.

TABLE 1/X.411

MTS-bind arguments

Argument	Presence	Clause
Bind arguments		
Initiator-name	M	8.1.1.1.1.1
Initiator-credentials	M	8.1.1.1.1.2
Security-context	O	8.1.1.1.1.3
Messages-waiting	O	8.1.1.1.1.4

8.1.1.1.1.1 *Initiator-name*

This argument contains a name for the initiator of the association. It shall be generated by the initiator of the association.

If the initiator is an MTS-user, the name is the **OR-name** of the MTS-user, which is registered with the MTS (see § 8.4.1.1.1.1). The **initiator-name** shall contain the **OR-address**, and may optionally also contain the **directory-name**, of the MTS-user (**OR-address-and-optional-directory-name**). For secure messaging, when an MS is involved, the **initiator-name** may also indicate whether the initiator is a UA or an MS.

If the initiator is the MTS (or an MTA – see § 11), the name is an **MTA-name**, which is known to the MTS-user.

8.1.1.1.1.2 *Initiator-credentials*

This argument contains the **credentials** of the initiator of the association. It shall be generated by the initiator of the association.

The **initiator-credentials** may be used by the responder to authenticate the identity of the initiator (see Recommendation X.509).

If only simple-authentication is used, the **initiator-credentials** comprise a simple **password** associated with the **initiator-name**.

If strong-authentication is used, the **initiator-credentials** comprise an **initiator-bind-token** and, optionally, an **initiator-certificate**.

The **initiator-bind-token** is a token generated by the initiator of the association. If the **initiator-bind-token** is an **asymmetric-token**, the **signed-data** comprises a **random number**. The **encrypted-data** of an **asymmetric-token** may be used to convey secret security-relevant information (e.g., one or more symmetric-encryption-keys) used to secure the association, or may be absent from the **initiator-bind-token**.

The **initiator-certificate** is a **certificate** of the initiator of the association, generated by a trusted source (e.g., a certification-authority). It may be supplied by the initiator of the association, if the **initiator-bind-token** is an **asymmetric-token**. The **initiator-certificate** may be used to convey a verified copy of the public-asymmetric-encryption-key (**subject-public-key**) of the initiator of the association. The initiator's public-asymmetric-encryption-key may be used by the responder to compute the **responder-bind-token**. If the responder is known to have, or have access to, the initiator's **certificate** (e.g., via the change-credentials abstract-operation, or via the directory), the **initiator-certificate** may be omitted.

8.1.1.1.1.3 *Security-context*

This argument identifies the **security-context** that the initiator of the association proposes to operate at. It may be generated by the initiator of the association.

The **security-context** comprises one or more **security-labels** that define the sensitivity of interactions that may occur between the MTS-user and the MTS for the duration of the association, in line with the security-policy in force. The **security-context** shall be one that is allowed by the registered **user-security-labels** of the MTS-user and by the **security-labels** associated with the MTA of the MTS.

Once established, the **security-context** of the submission-port and delivery-port can be temporarily restricted using the submission-control (see § 8.2.1.4.5) and delivery-control (see § 8.3.1.3.1.7) abstract-operation, respectively.

If **security-contexts** are not established between the MTS-user and the MTS, the sensitivity of interactions that may occur between the MTS-user and the MTS may be at the discretion of the invoker of an abstract-operation.

8.1.1.1.1.4 *Messages-waiting*

This argument indicates that the number of messages and total number of octets waiting to be delivered by the MTS to the MTS-user, for each **priority**. It may be generated by the initiator of the association.

This argument shall only be present when the MTS is initiating an association with an MTS-user, and when the MTS-user subscribes to the hold for delivery element-of-service (defined in Recommendation X.400).

8.1.1.1.2 *Results*

Table 2/X.411 lists the results of the MTS-bind, and for each result qualifies its presence and indicates the clause in which the result is defined.

TABLE 2/X.411

MTS-bind results

Result	Presence	Clause
Bind results		
Responder-name	M	8.1.1.1.2.1
Responder-credentials	M	8.1.1.1.2.2
Messages-waiting	O	8.1.1.1.2.3

8.1.1.1.2.1 *Responder-name*

This argument contains a name for the responder of the association. It shall be generated by the responder of the association.

If the responder is an MTS-user, the name is the **OR-name** of the MTS-user, which is registered with the MTS (see § 8.4.1.1.1.1). The **responder-name** shall contain the **OR-address**, and may optionally also contain the **directory-name**, of the MTS-user (**OR-address-and-optional-directory-name**). For secure messaging, when an MS is involved, the **responder-name** may also indicate whether the initiator is a UA or an MS.

If the responder is the MTS (or an MTA – see § 11), the name is an **MTA-name**, which is known to the MTS-user.

8.1.1.1.2.2 *Responder-credentials*

This argument contains the **credentials** of the responder of the association. It shall be generated by the responder of the association.

The **responder-credentials** may be used by the initiator to authenticate the identity of the responder (see Recommendation X.509).

If only simple-authentication is used, the **responder-credentials** comprise a simple **password** associated with the **responder-name**.

If strong-authentication is used, the **responder-credentials** comprise a **responder-bind-token**. The **responder-bind-token** is a **token** generated by the responder of the association. The **responder-bind-token** shall be the same type of **token** as the **initiator-bind-token**. If the **responder-bind-token** is an **asymmetric-token**, the **signed-data** comprises a **random-number** (which may be related to the **random-number** supplied in the **initiator-bind-token**). The **encrypted-data** of an **assymetric-token** may be used to convey secret security-relevant information (e.g., one or more symmetric-encryption-keys) used to secure the association, or may be absent from the **responder-bind-token**.

8.1.1.1.2.3 *Messages-waiting*

This argument indicates the number of messages and total number of octets waiting to be delivered by the MTS to the MTS-user, for each **priority**. It may be generated by the responder of the association.

This argument shall only be present when the MTS is responding to an association initiated by an MTS-user, and when the MTS-user subscribes to the hold for delivery element-of-service (defined in Recommendation X.400).

8.1.1.1.3 *Bind-errors*

The bind-errors that may disrupt the MTS-bind are defined in § 8.1.2.

8.1.1.2 *MTS-unbind*

The MTS-unbind enables the release of an established association by the initiator of the association.

8.1.1.2.1 *Arguments*

The MTS-unbind has no arguments.

8.1.1.2.2 *Results*

The MTS-unbind returns an empty result as indication of release of the association.

8.1.1.2.3 *Unbind-errors*

There are no unbind-errors that may disrupt the MTS-unbind.

8.1.2 *Bind-errors*

This section defines the following bind-errors:

a) Authentication-error
b) Busy
c) Unacceptable-dialogue-mode
d) Unacceptable-security-context.

8.1.2.1 *Authentication-error*

The authentication-error bind-error reports that an association cannot be established due to an authentication error; the initiator's **credentials** are not acceptable or are improperly specified.

The authentication-error bind-error has no parameters.

8.1.2.2 *Busy*

The busy bind-error reports that an association cannot be established because the responder is busy.

The busy-bind-error has no parameters.

8.1.2.3 *Unacceptable-dialogue-mode*

The unacceptable-dialogue-mode bind-error reports that the dialogue-mode proposed by the initiator of the association is unacceptable to the responder (see Recommendation X.419).

The unacceptable-dialogue-mode bind-error has no parameters.

8.1.2.4 *Unacceptable-security-context*

The unacceptable-security-context bind-error reports that the **security-context** proposed by the initiator of the association is unacceptable to the responder.

The unacceptable-security-context bind-error reports that the **security-context** proposed by the initiator of the association is unacceptable to the responder.

The unacceptable-security-context bind-error has no parameters.

8.2 *Submission port*

This section defines the abstract-operations and abstract-errors which occur at a submission-port.

8.2.1 *Abstract-operations*

This section defines the following submission-port abstract-operations.

a) Message-submission
b) Probe-submission
c) Cancel-deferred-delivery
d) Submission-control.

8.2.1.1 *Message-submission*

The message-submission abstract-operation enables an MTS-user to submit a message to the MTS for transfer and delivery to one or more recipient MTS-users.

The successful completion of the abstract-operation signifies that the MTS has accepted responsibility for the message (but not that it has yet delivered it to its intended recipients).

The disruption of the abstract-operation by an abstract-error indicates that the MTS cannot assume responsibility for the message.

8.2.1.1.1 *Arguments*

Table 3/X.411 lists the arguments of the message-submission abstract-operation, and for each argument qualifies its presence and identifies the clause in which the argument is defined.

8.2.1.1.1.1 *Originator-name*

This argument contains the **OR-name** of the originator of the message. It shall be generated by the originating MTS-user.

The **originator-name** contains the **OR-name** of an individual originator, i.e., it shall not contain the **OR-name** of a DL.

8.2.1.1.1.2 *Recipient-name*

This argument contains the **OR-name** of a recipient of the message. It shall be generated by the originator of the message. A different value of this argument shall be specified for each recipient of the message.

The **recipient-name** contains the **OR-name** of an individual recipient or DL.

8.2.1.1.1.3 *Alternate-recipient-allowed*

This argument indicates whether the message may be delivered to an alternate-recipient assigned by the recipient-MD, if the specified **recipient-name** does not identify an MTS-user. It may be generated by the originator of the message.

This argument may have one of the following values: **alternate-recipient-allowed** or **alternate-recipient-prohibited.**

TABLE 3/X.411

Message-submission arguments

Argument	Presence	Clause
Originator argument		
Originator-name	M	8.2.1.1.1.1
Recipient arguments		
Recipient-name	M	8.2.1.1.1.2
Alternate-recipient-allowed	O	8.2.1.1.1.3
Recipient-reassignment-prohibited	O	8.2.1.1.1.4
Originator-requested-alternate-recipient	O	8.2.1.1.1.5
DL-expansion-prohibited	O	8.2.1.1.1.6
Disclosure-of-recipients	O	8.2.1.1.1.7
Priority argument		
Priority	O	8.2.1.1.1.8
Conversion arguments		
Implicit-conversion-prohibited	O	8.2.1.1.1.9
Conversion-with-loss-prohibited	O	8.2.1.1.1.10
Explicit-conversion	O	8.2.1.1.1.11
Delivery time arguments		
Deferred-delivery-time	O	8.2.1.1.1.12
Latest-delivery-time	O	8.2.1.1.1.13
Delivery method argument		
Requested-delivery-method	O	8.2.1.1.1.14
Physical delivery arguments		
Physical-forwarding-prohibited	O	8.2.1.1.1.15
Physical-forwarding-address-request	O	8.2.1.1.1.16
Physical-delivery-modes	O	8.2.1.1.1.17
Registered-mail-type	O	8.2.1.1.1.18
Recipient-number-for-advice	O	8.2.1.1.1.19
Physical-rendition-attributes	O	8.2.1.1.1.20
Originator-return-address	O	8.2.1.1.1.21
Report request arguments		
Originator-report-request	M	8.2.1.1.1.22
Content-return-request	O	8.2.1.1.1.23
Physical-delivery-report-request	O	8.2.1.1.1.24
Security arguments		
Originator-certificate	O	8.2.1.1.1.25
Message-token	O	8.2.1.1.1.26
Content-confidentiality-algorithm-identifier	O	8.2.1.1.1.27
Content-integrity-check	O	8.2.1.1.1.28
Message-origin-authentication-check	O	8.2.1.1.1.29
Message-security-label	O	8.2.1.1.1.30
Proof-of-submission-request	O	8.2.1.1.1.31
Proof-of-delivery-request	O	8.2.1.1.1.32
Content arguments		
Original-encoded-information-types	O	8.2.1.1.1.33
Content-type	M	8.2.1.1.1.34
Content-identifier	O	8.2.1.1.1.35
Content-correlator	O	8.2.1.1.1.36
Content	M	8.2.1.1.1.37

If this argument has the value **alternate-recipient-allowed** and the **recipient-name** (specified by the originator of the message, or added by DL-expansion, or substituted by redirection to the **recipient-assigned-alternate-recipient** or the **originator-requested-alternate-recipient**, or present by any combination of redirection and expansion) does not identify an MTS-user, the message may be redirected to an alternate-recipient assigned by the recipient-MD to receive such messages. If no such alternate-recipient has been assigned by the recipient-MD, or if this argument has the value **alternate-recipient-prohibited**, a non-delivery report shall be generated.

In the absence of this argument, the default **alternate-recipient-prohibited** shall be assumed.

8.2.1.1.1.4 *Recipient-reassignment-prohibited*

This argument indicates whether the message may be reassigned to a **recipient-assigned-alternate-recipient** registered by the intended-recipient. It may be generated by the originator of the message.

This argument may have one of the following values: **recipient-reassignment-prohibited** or **recipient-reassignment-allowed**.

If this argument has the value **recipient-reassignment-allowed** and the intended-recipient has registered a **recipient-assigned-alternate-recipient**, the message shall be redirected to the **recipient-assigned-alternate-recipient**.

If this argument has the value **recipient-reassignment-prohibited** and the intended-recipient has registered a **recipient-assigned-alternate-recipient**, then if an **originator-requested-alternate-recipient** has been specified by the originator of the message, the message shall be redirected to the **originator-requested-alternate-recipient**, or if no **originator-requested-alternate-recipient** has been specified by the originator of the message, a non-delivery-report shall be generated.

In the absence of this argument, the default **recipient-reassignment-allowed** shall be assumed.

8.2.1.1.1.5 *Originator-requested-alternate-recipient*

This argument contains the **OR-name** of the alternate-recipient requested by the originator of the message. It may be generated by the originator of the message. A different value of this argument may be specified for each recipient of the message.

The **originator-requested-alternate-recipient** contains the **OR-name** of an individual, or DL, alternate-recipient.

If this argument is present and delivery of the message to the **recipient-name** (specified by the originator of the message, or added by DL-expansion, or substituted by redirection to the **originator-requested-alternate-recipient** specified by this argument.

If an **originator-requested-alternate-recipient** has been specified by the originator of the message, this message shall be redirected to that alternate recipient in preference to one assigned by the recipient-MD.

8.2.1.1.1.6 *DL-expansion-prohibited*

This argument indicates whether DL-expansion within an MTS shall occur for any **recipient-name** which denotes a DL. It may be generated by the originator of the message.

This argument may have one of the following values: **DL-expansion-prohibited** or **DL-expansion-allowed**.

In the absence of this argument, the default **DL-expansion-allowed** shall be assumed.

8.2.1.1.1.7 *Disclosure-of-recipients*

This argument indicates whether the **recipient-name** of all recipients are to be indicated to each recipient MTS-user when the message is delivered. It may be generated by the originator of the message.

This argument may have one of the following values: **disclosure-of-recipients-allowed** or **disclosure-of-recipients-prohibited**.

In the absence of this argument, the default **disclosure-of-recipients-prohibited** shall be assumed.

8.2.1.1.1.8 *Priority*

This argument specifies the relative priority of the message: **normal, non-urgent** or **urgent**. It may be generated by the originator of the message.

In the absence of this argument, a default **priority** of **normal** shall be assumed.

8.2.1.1.1.9 *Implicit-conversion-prohibited*

This argument indicates whether implicit-conversion may be performed on the message **content**. It may be generated by the originator of the message.

This argument may have one of the following values: **implicit-conversion-prohibited** or **implicit-conversion-allowed**.

In the absence of this argument, the default **implicit-conversion-allowed** shall be assumed.

See also § 8.2.1.1.1.10.

8.2.1.1.1.10 *Conversion-with-loss-prohibited*

This argument indicates whether **encoded-information-type** conversion(s) may be carried out on the message **content**, if such conversion(s) would result in loss of information. Loss of information is defined in Recommendation X.408. It may be generated by the originator of the message.

This argument may have one of the following values: **conversion-with-loss-prohibited** or **conversion-with-loss-allowed**.

In the absence of this argument, the default **conversion-with-loss-allowed** shall be assumed.

The combined effect of the **implicit-conversion-prohibited** and **conversion-with-loss-prohibited** arguments relate to implicit-conversion only and is defined in Table 4/X.411.

TABLE 4/X.411

Combined effect of conversion arguments

Implicit conversion	Conversion with loss	Combined effect
allowed	with-loss-allowed	allowed
allowed	with-loss-prohibited	with-loss-prohibited
prohibited	with-loss-allowed	prohibited
prohibited	with-loss-prohibited	prohibited

8.2.1.1.1.11 *Explicit-conversion*

This argument indicates the type of conversion of the message **content** explicitly requested by the originator for the recipient. It may be generated by the originator of the message. A different value of this argument may be specified for each recipient of the message.

This argument may have one of the following values: **no-explicit-conversion, ia5-text-to-teletex, teletex-to-telex, telex-to-ia5-text, telex-to-teletex, telex-to-g4-class-1, telex-to-videotex, ia5-text-to-telex, telex-to-g3-facsimile, ia5-text-to-g3-facsimile, ia5-text-to-g4-class-1, ia5-text-to-videotex, teletex-to-ia5-text, teletex-to-g3-facsimile, teletex-to-g4-class-1, teletex-to-videotex, videotex-to-telex, videotex-to-ia5-text,** or **videotex-to-teletex**. Other types of **explicit-conversion** may be defined by future versions of this Recommendation. **Explicit-conversion** shall be performed as specified in Recommendation X.408.

In the absence of this argument, the default **no-explicit conversion** shall be assumed.

Note – When specified for a recipient DL, **explicit-conversion** applies to all members of the DL.

8.2.1.1.1.12 *Deferred-delivery-time*

This argument specifies the **time** before which the message should not be delivered to the recipient(s). It may be generated by the originator of the message.

8.2.1.1.1.13 *Latest-delivery-time*

This argument contains the **time** after which the message should not be delivered to the recipient(s). It may be generated by the originator of the message.

The handling of non-delivery because of **latest-delivery-time** is described in § 14.3.2.4.

8.2.1.1.1.14 *Requested-delivery-method*

This argument indicates the requested method of delivery of the message to the recipient. It may be generated by the originator of the message. A different value of this argument may be specified for each recipient of the message.

This argument may have one or more of the following values: **any-delivery-method**, **mhs-delivery**, **physical-delivery**, **teletex-delivery**, **g3-facsimile-delivery**, **g4-facsimile-delivery**, **ia5-terminal-delivery**, **videotex-delivery** or **telephone-delivery**.

If more than one value of this argument is specified for a recipient, the sequence of the values shall be assumed to imply the originator's order of preference of delivery-methods.

In the absence of this argument, the default **any-delivery-method** shall be assumed.

If the **recipient-name** generated by the originator of the message contains a **directory-name** but omits an **OR-address**, the MTS may use the **requested-delivery-method** as an indication of which form of **OR-address** the **directory-name** should be mapped to by the MTS (e.g., using the Directory). If a form of **OR-address** appropriate to a **requested-delivery-method** cannot be found, a **recipient-improperly-specified** abstract error shall be returned to the originator of the message.

If the **recipient-name** generated by the originator of the message contains an **OR-address** of a form not appropriate to a **requested-delivery-method**, a non-delivery report shall be returned to the originator of the message.

If the originator-supplied **requested-delivery-method** conflicts with the recipient's preferred delivery-method (e.g., as registered in the Directory in the mhs-preferred-delivery-method attribute), the originator's **requested-delivery-method** takes precedence. If the originator's conversion requirements (see §§ 8.2.1.1.1.9 to 8.2.1.1.1.11), a non-delivery report shall be returned to the originator of the message.

8.2.1.1.1.15 *Physical-forwarding-prohibited*

This argument indicates whether physical-forwarding of the message is prohibited. It may be generated by the originator of the message if the **requested-delivery-method** argument specifies that physical-delivery is required to the recipient, or if the originator of the message supplied a **postal-OR-address** for the recipient. A different value of this argument may be specified for each recipient of the message.

This argument may have one of the following values: **physical-forwarding-allowed**, or **physical-forwarding-prohibited**.

In the absence of this argument, the default **physical-forwarding-allowed** shall be assumed.

8.2.1.1.1.16 *Physical-forwarding-address-request*

This argument indicates whether the physical-forwarding-address of the recipient is to be returned in this report. It may be generated by the originator of the message if the **requested-delivery-method** argument specifies that physical-delivery is required to the recipient, or if the originator of the message supplied a **postal-OR-address** for the recipient. A different value of this argument may be specified for each recipient of the message.

This argument may have one of the following values: **physical-forwarding-address-requested** or **physical-forwarding-address-not-requested**.

In the absence of this argument, the default **physical-forwarding-address-not-requested** shall be assumed.

A physical-forwarding-address may be requested when physical-forwarding is prohibited or allowed (see § 8.2.1.1.1.15).

8.2.1.1.1.17 *Physical-delivery-modes*

This argument indicates the mode of physical-delivery to the recipient to be used. It may be generated by the originator of the message if the **requested-delivery-method** argument specifies that physical-delivery is required to the recipient, or if the originator of the message supplied a **postal-OR-address** for the recipient. A different value of this argument may be specified for each recipient of the message.

This argument may have one of the following values: **ordinary-mail, special-delivery, express-mail, counter-collection, counter-collection-with-telephone-advice, counter-collection-with-telex-advice, counter-collection-with-teletex-advice,** or **bureau-fax-delivery.**

Note that **bureau-fax-delivery** comprises all A to H modes of delivery defined in Recommendation F.170, i.e., A – regular delivery, B – special delivery, C – express mail, D – counter collection, E – counter collection with telephone advice, F – telefax, G – counter collection with telex advice, and H – counter collection with teletex advice.

In the absence of this argument, the default **ordinary-mail** shall be assumed.

8.2.1.1.1.18 *Registered-mail-type*

This argument indicates the type of registered mail service to be used physically deliver the message to the recipient. It may be generated by the originator of the message if the **requested-delivery-method** argument specifies that physical delivery is required to the recipient, or if the originator of the message supplied a **postal-OR-address** for the recipient. A different value of this argument may be specified for each recipient of the message.

This argument may have one of the following values: **non-registered-mail, registered-mail,** or **registered-mail-to-addressee-in-person.**

In the absence of this argument, the default **ordinary-mail** shall be assumed.

8.2.1.1.1.19 *Recipient-number-for-advice*

This argument contains the telephone, telex or teletex number of the recipient, to be used in conjunction with the **counter-collection-with-advice** and **bureau-fax-delivery physical-delivery-modes.** It may be generated by the originator of the message if the **requested-delivery-method** argument specifies that physical-delivery is required to the recipient, or if the originator of the message supplied a **postal-OR-address** for the recipient, and the **physical-delivery-modes** argument specifies a **counter-collection-with-advice** or **bureau-fax-delivery physical-delivery-mode.** A different value of this argument may be specified for each recipient of the message.

8.2.1.1.1.20 *Physical-rendition-attributes*

This argument indicates the **physical-rendition-attributes** of the message. It may be generated by the originator of the message if the **requested-delivery-method** argument specifies that physical-delivery is required to the recipient, or if the originator of the message supplied a **postal-OR-address** for the recipient. A different value of this argument may be specified for each recipient of the message.

This argument may have one of the following values: **basic.** Future versions of this Recommendation may define other values of this argument. Other values of this argument may be used by bilateral agreement between MDs.

In the absence of this argument, the default **basic** shall be assumed.

8.2.1.1.1.21 *Originator-return-address*

This argument contains the **postal-OR-address** of the originator of the message. It shall be generated by the originator of the message if the **requested-delivery-method** argument specifies that physical-delivery is required to one or more recipients of the message, or if the originator of the message supplied one or more **postal-OR-address** for the recipients. It may also be generated by the originator of the message if a recipient DL contains, or is likely to contain, one or more members for whom physical-delivery is required.

The **originator-return-address** shall contain the **postal-OR-address** of an individual originator (**OR-address**), i.e., shall not contain the **directory-name** of an individual originator nor the **directory-name** of a DL.

8.2.1.1.1.22 *Originator-report-request*

This argument indicates the kind of report requested by the originator of the message. It shall be generated by the originator of the message. A different value of this argument may be specified for each recipient of the message.

This argument may have one of the following values:

- **no-report**: the originator of the message requested the suppression of non-delivery-reports;
- **non-delivery-report**: a report is returned only in case of non-delivery;
- **report**: a report is returned in case of delivery or non-delivery.

Note that the value of this argument may be changed at a DL expansion-point in line with the reporting-policy of the DL. Such a change may affect the number and type of reports the originator of the message may receive about delivery to a DL.

8.2.1.1.1.23 *Content-return-request*

This argument indicates whether the message **content** is to be returned with any non-delivery-report(s). It may be generated by the originator of the message.

This argument may have one of the following values: **content-return-requested** or **content-return-not-requested**.

In the absence of this argument, the default **content-return-not-requested** shall be assumed.

Note that the suppression of non-delivery-reports by the originator of the message (see § 8.2.1.1.1.22) takes precedence over a request for the return of the **content**.

Note that in the case of non-delivery-reports delivered to the owner of a DL (see § 8.3.1.2.1.4), the message **content** shall not be present.

8.2.1.1.1.24 *Physical-delivery-report-request*

This argument indicates the type of physical-delivery-report requested by the originator of the message. It may be generated by the originator of the message if the **requested-delivery-method** argument specifies that physical-delivery is required to the recipient or if the originator of the message supplied a **postal-OR-address** for the recipient. A different value of this argument may be specified for each recipient of the message.

This argument may have one of the following values: **return-of-undeliverable-mail-by-PDS, return-of-notification-by PDS, return-of-notification-by-MHS**, or **return-of-notification-by-MHS-and-PDS**.

In the absence of this argument, the default **return-of-undeliverable-mail-by-PDS** shall be assumed.

8.2.1.1.1.25 *Originator-certificate*

This argument contains the **certificate** of the originator of the message. It shall be generated by a trusted source (e.g. a certification-authority), and may be supplied by the originator of the message.

The **originator-certificate** may be used to convey a verified copy of the public-asymmetric-encryption-key (**subject-public-key**) of the originator of the message.

The originator's public-asymmetric-encryption-key may be used by the recipient(s) of the message to validate the **message-token**, if an **asymmetric-token**, if an **asymmetric-token** is used.

The originator's public-asymmetric-encryption-key may also be used by the recipient(s) of the message, and any MTA through which the message is transferred, to validate the **message-origin-authentication-check**.

8.2.1.1.1.26 *Message-token*

This argument contains the **token** associated with the message. It may be generated by the originator of the message. A different value of this argument may be specified for each recipient of the message.

If the **message-token** is an **asymmetric-token**, the **signed-data** may comprise:

- any of the following arguments: the **content-confidentiality-algorithm-identifier**, the **content-integrity-check**, the **message-security-label**, and the **proof-of-delivery-request**; and
- a **message-sequence-number**, that identifies the position of the message in a sequence of messages from the originator to the recipient to which the **message-token** relates (to provide the Message Sequence Integrity element-of-service, as defined in Recommendaton X.400).

If the **message-token** is an **asymmetric-token**, the **encrypted-data** may comprise:

- a **content-confidentiality-key**: a symmetric-encryption-key used with the **content-confidentiality-algorithm-identifier** by the originator of the message to encrypt the message **content**, and by the recipient to decrypt the message **content**; and/or
- the **content-integrity-check**: may be included in the **encrypted-data**, rather than the **signed-data**, if confidentiality of the **content-integrity-check** is required, and/or if the **message-security-label** is included in the **encrypted-data** (for confidentiality of the **message-security-label**) and the association between **content-integrity-check** and the **message-security-label** is to be maintained;
- the **message-security-label**: may be included in the **encrypted-data**, rather than the **signed-data**, if confidentiality of the **message-security-label** is required;
- a **content-integrity-key**: a symmetric-encryption-key used with the **content-integrity-algorithm-identifier** by the originator of the message to compute the **content-integrity-check**, and by the recipient to validate the **content-integrity-check**;
- a **message-sequence-number**: as defined for the **signed-data** above, but may be included in the **encrypted-data** instead if confidentiality of the sequence is required.

If the **message-token** is an **asymmetric-token** and the **signed-data** of the **message-token** includes the **content-integrity-check**, the **message-token** provides for non-repudiation-of-origin of message **content** (the non-repudiation of origin element-of-service, as defined in Recommendation X.400). If the **signed-data** of the **message-token** includes both the **content-integrity-check** and the **message-security-label**, the **message-token** provides proof of association between the **message-security-label** and the message **content**.

8.2.1.1.1.27 *Content-confidentiality-algorithm-identifier*

This argument contains an **algorithm-identifier**, which identifies the algorithm used by the originator of the message to encrypt the message **content** (to provide the content confidentiality element-of-service as defined in Recommendation X.400). It may be generated by the originator of the message.

The algorithm may be used by the recipient(s) of the message to decrypt the message **content**.

The content-confidentiality altorithm may be either a symmetric- or an asymmetric-encryption-algorithm.

If a symmetric-encryption-algorithm is used, the **content-confidentiality-key** used by the originator to encrypt the message **content**, and which the recipient may use to decrypt the message **content**, may be derived from the **message-token** sent with the message. Alternatively, **content-confidentiality-key** may be distributed by some other means.

If an asymmetric-encryption-algorithm is used, the intended-recipient's public-asymmetric-encryption-key may be used by the originator of the message to encrypt the message **content**. The recipient may use the recipient's secret-asymmetric-encryption-key to decrypt the message **content**. Note that if an asymmetric-encryption-algorithm is used, the message can only be addressed to a single recipient, or to a set of recipients which share the same asymmetric-encryption-key pair.

8.2.1.1.1.28 *Content-integrity-check*

This argument provides the recipient(s) of the message with a means of validating that the message **content** has not been modified (to provide the content integrity element-of-service as defined in Recommendation X.400). It may be generated by the originator of the message. A different value of the argument may be specified for each recipient of the message.

The **content-integrity-check** enables content-integrity to be validated on a per-recipient basis using either a symmetric- or an asymmetric-encryption-algorithm. Note that the **message-origin-authentication-check** provides a means of validating content-integrity on a per-message basis using an asymmetric-encryption-algorithm.

The **content-integrity-check** may be included in the **signed-data** or the **encrypted-data** of the **message-token** to provide for non-repudiation-of-origin of the message **content**, and proof of association between the **message-security-label** and the message **content**.

The **content-integrity-check** is computed using the algorithm identified by the **content-integrity-algorithm-identifier** (an **algorithm-identifier**).

The **content-integrity-check** contains the **content-integrity-algorithm-identifier**, and an encrypted function (e.g., a compressed or hashed version) of the **content-integrity-algorithm-identifier** and the message **content**. Note that the **content-integrity-check** is computed using the clear (i.e. unencrypted) message **content**.

The content-integrity-algorithm may be either a symmetric- or an asymmetric-encryption-algorithm. Note that the use of a symmetric-encryption algorithm may permit simultaneous compression and encryption of the message **content**.

If a symmetric-encryption-algorithm is used, the **content-integrity-key** used to compute the **content-integrity-check**, and which the recipient may use to validate the **content-integrity-check**, may be derived from the **message-token** sent with the message. Alternatively, the **content-integrity-key** may be distributed by some other means.

If an asymmetric-encryption-algorithm is used, the originator's secret-asymmetric-encryption-key may be used by the originator of the message to compute the **content-integrity-check**. The recipient may use the originator's public-asymmetric-encryption-key (**subject-public-key**) derived from the **originator-certificate** to validate the **content-integrity-check**.

8.2.1.1.1.29 *Message-origin-authentication-check*

This argument provides the recipient(s) of the message, and any MTA through which the message is transferred, with a means of authenticating the origin of the message (to provide the Message Origin Authentication element-of-service as defined in Recommendation X.400). It may be generated by the originator of the message.

The **message-origin-authentication-check** provides proof of the origin of the message (message origin authentication), assurance that the message **content** has not been modified (the content integrity element-of-service as defined in Recommendation X.400), and proof of association between the **message-security-label** and the message.

The **message-origin-authentication-check** is computed using the algorithm (asymmetric-encryption-algorithm and hash-function) identified by the **message-origin-authentication-algorithm-identifier** (an **algorithm-identifier**).

The **message-origin-authentication-check** contains the **message-origin-authentication-algorithm-identifier**, and an asymmetrically encrypted, hashed version of the **message-origin-authentication-algorithm-identifier**, the message **content**, the **content-identifier** and the **message-security-label**. Optional components are included in the **message-origin-authentication-check** if they are present in the message.

If content-confidentiality (see § 8.2.1.1.1.27) is also used, the **message-origin-authentication-check** is computed using the encrypted version of the message **content** (to allow the **message-origin-authentication-check** to be validated by other than the intended-recipient (e.g. by an MTA) without compromising the confidentiality of the message **content**). Note that if the clear (i.e. unencrypted) version of the message **content** is used to compute the **message-origin-authentication-check**, the **message-origin-authentication-check** provides for both message-origin authentication and non-repudiation of origin of the message **content** (a signature), as defined in Recommendation X.400. If, however, the encrypted version of the message **content** is used, the **message-origin-authentication-check** provides for message-origin authentication, but not for non-repudiation of origin of the message **content**.

The **message-origin-authentication-check** may be computed by the originator of the message using the originator's secret-asymmetric-encryption-key. The **message-origin-authentication-check** may be validated by the recipient(s) of the message, and any MTA through which the message is transferred, using the public-asymmetric-encryption-key (**subject-public-key**) of the originator of the message derived from the **originator-certificate**.

Future version of this Recommendation may define other forms of **message-origin-authentication-check** (e.g., based on symmetric-encryption-techniques) which may be used by MTAs through which the message is transferred to authenticate the origin of the message.

8.2.1.1.1.30 *Message-security label*

This argument associates a **security-label** with the message (or probe). It may be generated by the originator of the message (or probe), in line with the security-policy in force.

The **message-security label** of a report shall be the same as the **message-security label** of the subject-message (or subject-probe).

If **security-labels** are assigned to MTS-users, MTAs and other objects in the MHS, the handling by those objects of messages, probes and reports bearing **message-security-labels** may be determined by the security-policy in force. If **security-labels** are not assigned to MTS-users, MTAs and other objects in the MHS, the handling by those objects of messages, probes and reports bearing **message-security-labels** may be discretionary.

If **security-contexts** are established between the originator and an MTA (the originating-MTA) of the MTS (see §§ 8.1.1.1.1.3 and 8.2.1.4.1.5), the **message-security-label** that the originator may assign to a message (or probe) may be determined by the **security-context** (submission-security-context), in line with the security-policy in force. If **security-contexts** are not established between the originator and the originating-MTA, the assignment of a **message-security-label** to a message (or probe) may be at the discretion of the originator.

If **security-contexts** are established between two MTAs (see § 12.1.1.1.1.3), the transfer of messages, probes or reports between the MTAs may be determined by the **message-security-labels** of the messages, probes or reports, and the **security-context**, in line with the security-policy in force. If **security-contexts** are not established between the MTAs, the transfer of messages, probes and reports may be at the discretion of the sender.

If **security-contexts** are established between an MTS-user and an MTA (the delivering-MTA) of the MTS (see §§ 8.1.1.1.1.3 and 8.3.1.3.1.7), the delivery of messages and reports may be determined by the **message-security-labels** of the messages and reports, and the **security-context** (delivery-security-context), in line with the security-policy in force. If the **message-security-label** of a message or report is allowed by the registered **user-security-labels** of the recipient, but disallowed by the recipient's current **security-context** (delivery-security-context), then the delivering-MTA may hold-for-delivery. If **security-contexts** are not established between the MTS-user and the delivering-MTA, the delivery of messages and reports may be at the discretion of the delivering-MTA.

8.2.1.1.1.31 *Proof-of-submission-request*

This argument indicates whether or not the originator of the message requires **proof-of-submission** (to provide the proof of submission element-of-service) as defined in Recommendation X.400) of the message to the MTS. It may be generated by the originator of the message.

This argument may have one of the following values: **proof-of-submission-requested** or **proof-of-submission-not-requested**.

In the absence of this argument, the default **proof-of-submission-not-requested** shall be assumed.

8.2.1.1.1.32 *Proof-of-delivery-request*

This argument indicates whether or not the originator of the message requires **proof-of-delivery** (to provide the proof of delivery element-of-service as defined in Recommendation X.400) of the message to the recipient. It may be generated by the originator of the message. A different value of this argument may be specified for each recipient of the message.

This argument may have one of the following values: **proof-of-delivery-requested** or **proof-of-delivery-not-requested**.

In the absence of this argument, the default **proof-of-delivery-not-requested** shall be assumed.

8.2.1.1.1.33 *Original-encoded-information-types*

This argument identifies the original **encoded-information-types** of the message **content**. It may be generated by the originator of the message.

The absence of this argument indicates that the **original-encoded-information-types** of the message **content** are unspecified.

8.2.1.1.1.34 *Content-type*

This argument identifies the type of the content of the message. It shall be generated by the originator of the message. The **content-type** shall be either built-in or extended.

A built-in **content-type** may have one of the following values:

- **unidentified**: denotes a **content-type** unidentified and unconstrained; the of this **unidentified content-type** is by bilateral agreement between MTS-users;
- **external**: denotes a **content-type** which is reserved for use when interworking between 1988 systems and 1984 systems (see Recommendation X.419);
- **interpersonal-messaging-1984**: identifies the **interpersonal-messaging-1984 content-type** defined in Recommendation X.420;

- **interpersonal-messaging-1988**: identifies the **interpersonal-messaging-1988 content-type** defined in Recommendation X.420;
- one specific value of an extended **content-type** which has been defined by this Recommendation is **inner-envelope**: an extended **content-type** that is itself a message (envelope and content), for forwarding by the recipient named on the outer-envelope to those named on the inner-envelope. The type of the **content** OCTET STRING in an **MTS-APDU**, encoded using the Basic Encoding Rules of ASN.1. [Note that the inner-envelope and using the security arguments (see §§ 8.2.1.1.1.25 to 8.2.1.1.1.32).]

Other standardized **extended content-types** may be defined by future versions of this Recommendation. Other values of this argument may be used by bilateral agreement between MTS-users.

8.2.1.1.1.35 *Content-identifier*

This argument contains an identifier for the **content** of the message. It may be generated by the originator of the message.

The **content-identifier** may be delivered to the recipient(s) of the message, and is returned to the originator with any report(s). This argument is not altered by the MTS.

8.2.1.1.1.36 *Content-correlator*

This argument contains information to enable correlation of the **content** of the message by the originator of the message. It may be generated by the originator of the message.

The **content-correlator** is not delivered to the recipient(s) of the message, but is returned to the originator with any report(s). This argument is not altered by the MTS.

8.2.1.1.1.37 *Content*

This argument contains the information the message is intended to convey to the recipient(s). It shall be generated by the originator of the message.

Except when conversion is performed, the **content** of the message is not modified by the MTS, but rather is passed transparently through it.

The **content** may be encrypted to ensure its confidentiality (see § 8.2.1.1.1.27).

The **content** may be an **external-content**. The **content** is an **external-content** when the **content-type** argument has the value **external**. When the **content** is an **external-content**, the **external-content-type** is specified by the object identifier of the **external-content**. An **external-content** may be used to convey an **inner-envelope** (see § 8.2.1.1.1.34), or for interworking between 1988 systems and 1984 systems (see Recommendation X.419).

8.2.1.1.2 *Results*

Table 5/X.411 lists the results of the message-submission abstract-operation, and for each result qualifies its presence and identifies the clause in which the result is defined.

TABLE 5/X.411

Message-submission results

Result	Presence	Clause
Message-submission-identifier	M	8.2.1.1.2.1
Message-submission-time	M	8.2.1.1.2.2
Originating-MTA-certificate	O	8.2.1.1.2.3
Proof-of-submission	C	8.2.1.1.2.4
Content-identifier	C	8.2.1.1.1.35

8.2.1.1.2.1 *Message-submission-identifier*

This result contains an **MTS-identifier** that uniquely and unambiguously identifies the message-submission. It shall be generated by the MTS.

The MTS provides the **message-submission-identifier** when notifying the MTS-user, via the report-delivery abstract-operation, of the delivery or non-delivery of the message.

The MTS-user provides the **message-submission-identifier** when cancelling, via the cancel-deferred-delivery abstract-operation, a message whose delivery it deferred.

8.2.1.1.2.2 *Message-submission-time*

This result indicates the **time** at which the MTS accepts responsibility for the message. It shall be generated by the MTS.

8.2.1.1.2.3 *Originating-MTA-certificate*

This result contains the **certificate** of the MTA to which the message has been submitted (the originating-MTA). It shall be generated by a trusted source (e.g. a certification-authority), and may be supplied by the originating-MTA, if the originator of the message requested **proof-of-submission** (see § 8.2.1.1.1.31) and an asymmetric-encryption-algorithm is used to compute the **proof-of-submission.**

The **originating-MTA-certificate** may be used to convey to the originator of the message a verified copy of the public-asymmetric-encryption-key (**subject-public-key**) of the originating MTA.

The originating-MTA's public-asymmetric-encryption-key may be used by the originator of the message to validate the **proof-of-submission.**

8.2.1.1.2.4 *Proof-of-submission*

This result provides the originator of the message with proof of submission of the message to the MTS (to provide the proof of submission element-of-service as defined in Recommendation X.400). Depending on the encryption-algorithm used and the security policy in force, this argument may also provide the non-repudiation of submission element-of-service (as defined in Recommendation X.400). It shall be generated by the originating-MTA of the MTS, if the originator of the message requested **proof-of-submission** (see § 8.2.1.1.1.31).

The **proof-of-submission** is computed using the algorithm identified by the **proof-of-submission-algorithm-identifier** (an **algorithm-identifier.**)

The **proof-of-submission** contains the **proof-of-submission-algorithm-identifier**, and an encrypted function (e.g., a compressed or hashed version) of the **proof-of-submission-algorithm-identifier**, the arguments of the submitted message (see § 8.2.1.1.1), and the **message-submission-identifier** and **message-submission-time**. Optional components are included in the **proof-of-submission** if they are present in the message.

Note that receipt of this result provides the originator of the message with proof of submission of the message. Non-receipt of this result provides neither proof of submission nor proof of non-submission (unless a secure link and trusted functionality are employed).

If an asymmetric-encryption-algorithm is used, the **proof-of-submission** may be computed by the originating-MTA using the originating-MTA's secret-asymmetric-encryption-key. The originator of the message may validate the **proof-of-submission** using the originating-MTA's public-asymmetric-encryption-key (**subject-public-key**) derived from the **originating-MTA-certificate**. An asymmetric **proof-of-submission** may also provide for non-repudiation of submission.

If a symmetric-encryption-algorithm is used, the summetric-encryption-key that the originating-MTA used to compute the **proof-of-submission**, and which the originator may use to validate the **proof-of-submission**, may be derived from the **bind-tokens** (see §§ 8.1.1.1.1.3 and 8.1.1.1.2.2) exchanged when the association was initiated. Alternatively, the symmetric-encryption-key used for **proof-of-submission** may be exchanged by some other means. Note that if a symmetric-encryption-algorithm is used then the **proof-of-submission** can only support non-repudiation of submission if the security-policy in force provides for the involvement of a third party acting as a notary.

8.2.1.1.3 *Abstract-errors*

Table 6/X.411 lists the abstract-errorsthat may disrupt the message-submission abstract-operation, and for each abstract-error identifies the clause in which the abstract-error is defined.

TABLE 6/X.411

Message-submission abstract-errors

Abstract-error	Clause
Submission-control-violated	8.2.2.1
Element-of-service-not-subscribed	8.2.2.2
Originator-invalid	8.2.2.4
Recipient-improperly-specified	8.2.2.5
Inconsistent-request	8.2.2.7
Security-error	8.2.2.8
Unsupported-critical-function	8.2.2.9
Remote-bind-error	8.2.2.10

8.2.1.2 *Probe-submission*

The probe-submission abstract-operation enables an MTS-user to submit a probe in order to determine whether or not a message (the subject-message) could be transferred and delivered to one or more recipient MTS-users if it were to be submitted.

Success of a probe does not guarantee that a subsequently submitted message can actually be delivered but rather that, currently, the recipient is valid and the message would encounter no major obstacles to delivery.

For any **recipient-names** that denote a DL, the probe-submission abstract-operation determines whether expansion of the specified DL (but not of any nested DLs) would occur.

For any **recipient-names** for which redirection would occur, the probe-submission abstract-operation determines whether the message could be transferred and delivered to the alternate-recipient.

The MTS-user supplies most of the arguments used for message-submission and the length of the content of the subject-message. The probe-submission abstract-operation does not culminate in delivery to the intended recipients of the subject-message, but establishes whether or not the message-submission abstract-operation would be likely to do so.

The successful completion of the abstract-operation signifies that the MTS has agreed to undertake the probe (but not that it has yet performed the probe).

The disruption of the abstract-operation by an abstract-error indicates that the MTS cannot undertake the probe.

8.2.1.2.1 *Arguments*

Table 7/X.411 lists the arguments of the probe-submission abstract-operation, and for each argument qualifies its presence and identifies the clause in which the argument is defined.

8.2.1.2.1.1 *Probe-origin-authentication-check*

This argument provides any MTA through which the probe is transferred, with a means of authenticating the origin of the probe (to provide the probe origin authentication element-ofservice as defined in Recommendation X.400). It may be generated by the originator of the probe.

The **probe-origin-authentication-check** provides proof of the origin of the probe (Probe Origin Authentication), and proof of association between the **message-security-label** and the **content-identifier** of the subject-message.

The **probe-origin-authentication-check** is computed using the algorithm identified by the **probe-origin-authentication-algorithm-identifier** (an **algorithm-identifier**).

TABLE 7/X.411

Probe-submission arguments

Argument	Presence	Clause
Originator argument		
Originator-name	M	8.2.1.1.1.1
Recipient arguments		
Recipient-name	M	8.2.1.1.1.2
Alternate-recipient-allowed	O	8.2.1.1.1.3
Recipient-reassignment-prohibited	O	8.2.1.1.1.4
Originator-requested-alternate-recipient	O	8.2.1.1.1.5
DL-expansion-prohibited	O	8.2.1.1.1.6
Conversion arguments		
Implicit-conversion-prohibited	O	8.2.1.1.1.9
Conversion-with-loss-prohibited	O	8.2.1.1.1.10
Explicit-conversion	O	8.2.1.1.1.11
Delivery method argument		
Requested-delivery-method	O	8.2.1.1.1.14
Physical delivery argument		
Physical-rendition-attributes	O	8.2.1.1.1.20
Report request argument		
Originator-report-request	M	8.2.1.1.1.22
Security arguments		
Originator-certificate	O	8.2.1.1.1.25
Probe-origin-authentication-check	O	8.2.1.2.1.1
Message-security-label	O	8.2.1.1.1.30
Content arguments		
Original-encoded-information-types	O	8.2.1.1.1.33
Content-type	M	8.2.1.1.1.34
Content-identifier	O	8.2.1.1.1.35
Content-correlator	O	8.2.1.1.1.36
Content-length	O	8.2.1.2.1.2

The **probe-origin-authentication-check** contains the **probe-origin-authentication-algorithm-identifier**, and an asymmetrically encrypted, hashed version of the **probe-origin-authentication-algorithm-identifier**, and the **content-identifier** and **message-security-label** of the subject-message. Optional components are included in the **probe-origin-authentication-check** if they are present in the probe.

Future versions of this Recommendation may define other forms of **probe-origin-authentication-check** (e.g., based on symmetric-encryption-techniques) which may be used by MTAs through which the probe is transferred to authenticate the origin of the probe.

8.2.1.2.1.2 *Content-length*

This argument specifies the length, in octets, of the **content** of the subject-message. It may be generated by the originator of the probe.

8.2.1.2.2 *Results*

Table 8/X.411 lists the results of the probe-submission abstract-operation, and for each result qualifies its presence and identifies the clause in which the result is defined.

TABLE 8/X.411

Probe-submission results

Result	Presence	Clause
Probe-submission-identifier	M	8.2.1.2.2.1
Probe-submission-time	M	8.2.1.2.2.2
Content-identifier	C	8.2.1.1.1.35

8.2.1.2.2.1 *Probe-submission-identifier*

This result contains an **MTS-identifier** that uniquely and unambiguously identifies the probe-submission. It shall be generated by the MTS.

The MTS provides the **probe-submission-identifier** when notifying the MTS-user, via the report-delivery abstract-operation, of its ability or otherwise to deliver the subject-message.

8.2.1.2.2.2 *Probe-submission-time*

This result indicates the **time** at which the MTS agreed to undertake the probe. It shall be generated by the MTS.

8.2.1.2.3 *Abstract-errors*

Table 9/X.411 lists the abstract-errors that may disrupt the probe-submission abstract-operation, and for each abstract-error identifies the clause in which the abstract-error is defined.

TABLE 9/X.411

Probe-submission abstract-errors

Abstract-error	Clause
Submission-control-violated	8.2.2.1
Element-of-service-not-subscribed	8.2.2.2
Originator-invalid	8.2.2.4
Recipient-improperly-specified	8.2.2.5
Inconsistent-request	8.2.2.7
Security-error	8.2.2.8
Unsupported-critical-function	8.2.2.9
Remote-bind-error	8.2.2.10

8.2.1.3 *Cancel-deferred-delivery*

The cancel-deferred-delivery abstract-operation enables an MTS-user to abort the deferred-delivery of a message previsouly submitted via the message-submission abstract-operation.

The MTS-user identifies the message whose delivery is to be cancelled by means of the **message-submission-identifier** returned by the MTS as a result of the previous invocation of the message-submission abstract-operation.

The successful completion of the abstract-operation signifies that the MTS has cancelled the deferred-delivery of the message.

The disruption of the abstract-operation by an abstrar-error indicates that the deferred-delivery cannot be cancelled. The deferred-delivery of a message cannot be cancelled if the message has already been progressed for delivery and/or transfer within the MTS. The MTS may refuse to cancel the deferred-delivery of a message if the MTS provided the originator of the message with **proof-of-submission.**

8.2.1.3.1 *Arguments*

Table 10/X.411 lists the arguments of the cancel-deferred-delivery abstract-operation, and for each argument qualifies its presence and identifies the clause in which the argument is defined.

TABLE 10/X.411

Cancel-deferred-delivery arguments

Argument	Presence	Clause
Submission argument		
Message-submission-identifier	M	8.2.1.3.1.1

8.2.1.3.1.1 *Message-submission-identifier*

This argument contains the **message-submission-identifier** of the message whose deferred-delivery is to be cancelled. It shall be suppled by the MTS-user.

The **message-submission-identifier** (an **MTS-identifier**) is that returned by the MTS as a result of a previous invocation of the message-submission abstract-operation (see § 8.2.1.1.2.1), when the message was submitted for deferred-delivery.

8.2.1.3.2 *Results*

The cancel-deferred-delivery abstract-operation returns an emply result as indication of success.

8.2.1.3.3 *Abstract-errors*

Table 11/X.411 lists the abstract-errors that may disrupt the cancel-deferred-delivery abstract-operation, and for each abstract-error identifies the clause in which the abstract-error is defined.

TABLE 11/X.411

Cancel-deferred-delivery abstract-errors

Abstract-error	Clause
Deferred-delivery-cancellation-rejected	8.2.2.3
Message-submission-identifier-invalid	8.2.2.6
Remote-bind-error	8.2.2.10

8.2.1.4 *Submission-control*

The submission-control abstract-operation enables the MTS to temporarily limit the submission-port abstract-operations that the MTS-user may invoke, and the messages that the MTS-user may submit to the MTS via the Message-submission abstract-operation.

The MTS-user should hold until a later time, rather than abandon, abstract-operations and messages presently forbidden.

The successful completion of the abstract-operation signifies that the specified controls are now in force. These controls supersede any previously in force, and remain in effect until the association is released or the MTS re-invokes the submission-control abstract-operation.

The abstract-operation returns and indication of any abstract-operations that the MTS-user would invoke, or any message types that the MTS-user would submit, were it not for the prevailing controls.

8.2.1.4.1 *Arguments*

Table 12/X.411 lists the arguments of the submission-control abstract-operation, and for each argument qualifies its presence and identifies the clause in which the argument is defined.

TABLE 12/X.411

Submission-control arguments

Argument	Presence	Clause
Submission control arguments		
Restrict	O	8.2.1.4.1.1
Permissible-operations	O	8.2.1.4.1.2
Permissible-lowest-priority	O	8.2.1.4.1.3
Permissible-maximum-content-length	O	8.2.1.4.1.4
Permissible-security-context	O	8.2.1.4.1.5

8.2.1.4.1.1 *Restrict*

This argument indicates whether the controls on submission-port abstract-operations are to be updates or removed. It may be generated by the MTS.

This argument may have one of the following values:

- **update**: the other arguments update the prevailing controls;
- **remove**: all controls are to be removed; the other arguments are to be ignored.

In the absence of this argument, the defauld **update** shall be assumed.

8.2.1.4.1.2 *Permissible-operations*

This argument indicates the abstract-operations that the MTS-user may invoke on the MTS.

This argument may have the value **allowed** or **prohibited** for each of the following:

- **message-submission**: the MTS-user may/may not invoke the message-submission abstract-operation; and
- **probe-submission**: the MTS-user may/may not invoke the probe-submission abstract-operation.

Other submission-port abstract-operations are not subject to controls, and may be invoked at any time.

In the absence of this argument, the abstract-operation that the MTS-user may invoke on the MTS are unchanged. If no previous controls are in force, the MTS-user may invoke both the message-submission abstract-operation and the probe-submission abstract-operation.

8.2.1.4.1.3 *Permissible-lowest-priority*

This argument contains the **priority** of the lowest priority message that the MTS-user shall submit to the MTS via the message-submission abstract-operation. It may be generated by the MTS.

This argument may have one of the following values of the **priority** argument of the message-submission abstract-operation: **normal, non-urgent** or **urgent**

In the absence of this argument, the **priority** of the lowest priority message that the MTS-user shall submit to the MTS is unchanged. If no previous controls are in force, the MTS-user may submit messages of any priority.

8.2.1.4.1.4 *Permissible-maximum-content-length*

This argument contains the **content-lenght**, in octets, of the longest-content message that the MTS-user shall submit to the MTS via the message-submission abstract-operation. It may be generated by the MTS.

In the absence of this argument, the **permissible-maximum-content-length** of a message that the MTS-user may submit to the MTS is unchanged. If no previous controls are in force, the content length is not explicitly limited.

8.2.1.4.1.5 *Permissible-security-context*

This argument temporarily limits the sensitivity of submission-port abstract-operations (submission-security-context) that the MTS-user may invoke on the MTS. It is a temporary restriction of the **security-context** established when the association was initiated (see § 8.1.1.1.1.3). It may be generated by the MTS.

The **permissible-security-context** comprises one or more **security-labels** from the set of **security-labels** established as the **security-context** when the association was established.

In the absence of this argument, the **security-context** of submission-port abstract-operations is unchanged.

8.2.1.4.2 *Results*

Table 13/X.411 lists the results of the submission-control abstract-operation, and for each result qualifies its presence and identifies the clause in which the result is defined.

TABLE 13/X.411

Submission-control results

Result	Presence	Clause
"Waiting" results		
Waiting-operations	O	8.2.1.4.2.1
Waiting-messages	O	8.2.1.4.2.2
Waiting-encoded-information-types	O	8.2.1.4.2.3
Waiting-content-types	O	8.2.1.4.2.4

8.2.1.4.2.1 *Waiting-operations*

This result indicates the abstract-operations being held by the MTS-user, and that the MTS-user would invoke on the MTS if it were not for the prevailing controls. It may be generated by the MTS-user.

This result may have the value **holding** or **not-holding** for each of the following:

- **message-submission**: the MTS user is/is not holding messages, and would invoke the message-submission abstract-operation on the MTS if it were not for the prevailing controls; and
- **probe-submission**: the MTS-user is/is not holding probes, and would invoke the probe-submission abstract-operation on the MTS if it were not for the prevailing controls.

In the absence of this result, it may be assumed that the MTS-user is not holding any messages or probes for submission to the MTS due to the prevailing controls.

8.2.1.4.2.2 *Waiting-messages*

This result indicates the kind of messages the MTS-user is holding for submission to the MTS, and would submit via the message-submission abstract-operation, if it were not for the prevailing controls. It may be generated by the MTS-user.

This result may have one or more of the following values:

- **long-content**: the MTS-user has messages held for submission to the MTS which exceed the **permissible-maximum-content-length** control currently in force;
- **low-priority**: the MTS-user has messages held for submission to the MTS of a lower **priority** than the **permissible-lowest-priority** control currently in force;
- **other-security-labels**: the MTS-user has messages held for submission to the MTS bearing **message-security-labels** other than those permitted by the current security-context.

In the absence of this result, it may be assumed that the MTS-user is not holding any messages or probes for submission to the MTS due to the **permissible-maximum-content-length**, **permissible-lowest-priority** or **permissible-security-context** controls currently in force.

8.2.1.4.2.3 *Waiting-encoded-information-types*

This result indicates the **encoded-information-types in the content** of any messages held by the MTS-user for submission to the MTS due to prevailing controls. It may be generated by the MTS-user.

In the absence of this result, the **encoded-information-types** of any messages held by the MTS-user for submission to the MTS are **unspecified.**

8.2.1.4.2.4 *Waiting-content-types*

This result indicates the **content-types** of any messages held by the MTS-user for submission to the MTS due to prevailing controls. It may be generated by the MTS-user.

In the absence of this result, the **content-types** of any messages held by the MTS-user for submission to the MTS are unspecified.

8.2.1.4.3 *Abstract-errors*

Table 14/X.411 lists the abstract-errors that may disrupt the submission-control abstract-operation, and for each abstract-error identifies the clause in which the abstract-error is defined.

TABLE 14/X.411

Submission-control abstract-errors

Abstract-error	Clause
Security-error	8.2.2.8
Remote-bind-error	8.2.2.10

8.2.2 *Abstract-errors*

This section defines the following submission-port abstract-errors:

a) submission-control-violated
b) element-of-service-not-subscribed
c) deferred-delivery-cancellation-rejected
d) originator-invalid
e) recipient-improperly-specified
f) message-submission-identifier-invalid
g) inconsistent-request
h) security-error
i) unsupported-critical-function
j) remote-bind-error.

8.2.2.1 *Submission-control-violated*

The submission-control-violated abstract-error reports the violation by the MTS-user of a control on submission-port services imposed by the MTS via the submission-control service.

The submission-control-violated abstract-error has no parameters.

8.2.2.2 *Element-of-service-not-subscribed*

The element-of-service-not-subscribed service reports that the requested abstract-operation cannot be provided by the MTS because the MTS-user has not subscribed to one of the elements-of-service the request requires.

The element-of-service-not-subscribed abstract-error has no parameters.

8.2.2.3 *Deferred-delivery-cancellation-rejected*

The deferred-delivery-cancellation-rejected abstract-error reports that the MTS cannot cancel the deferred-delivery of a message, either because the message has already been progressed for transfer and/or delivery, or because the MTS had provided the originator with **proof-of-submission**.

The deferred-delivery-cancellation-rejected abstract-error has no parameters.

8.2.2.4 *Originator-invalid*

The originator-invalid abstract-error reports that the message or probe cannot be submitted because the originator is incorrectly identified.

The originator-invalid abstract-error has no parameters.

8.2.2.5 *Recipient-improperly-specified*

The recipient-improperly-specified abstract-error reports that the message or probe cannot be submitted because one or more recipients are improperly specified.

The recipient-improperly-specified abstract-error has the following parameters, generated by the MTS.

– **improperly-specified-recipients**: the improperly specified **recipient-name(s)**.

8.2.2.6 *Message-submission-identifier-invalid*

The message-submission-identifier-invalid abstract-error reports that the deferred-delivery of a message cannot be cancelled because the specified **message-submission-identifier** is invalid.

The message-submission-identifier-invalid abstract-error has no parameters.

8.2.2.7 *Inconsistent-request*

The inconsistent-request abstract-error reports that the requested abstract-operation cannot be provided by the MTS because the MTS-user has made an inconsistent request.

The inconsistent-request abstract-error has no parameters.

8.2.2.8 *Security-error*

The security-error abstract-error reports that the requested abstract-operation could not be provided by the MTS because it would violate the security-policy in force.

The security-error abstract-error has the following parameters, generated by the MTS:

– **security-problem**: an identifier for the cause of the violation of the security-policy.

8.2.2.9 *Unsupported-critical-function*

The unsupported-critical-function abstract-error reports that an argument of the abstract-operation was marked as **critical-for-submission** (see § 9.1) but is unsupported by the MTS.

The unsupported-critical-function abstract-error has no parameters.

8.2.2.10 *Remote-bind-error*

The remote-bind-error abstract-error reports that the requested abstract-operation cannot be provided by the MS because the MS is unable to bind to the MTS. Note that this abstract-error only occurs on indirect submission to the MTS via an MS.

The remote-bind-error abstract-error has no parameters.

8.3 *Delivery port*

This paragraph defines the abstract-operations and abstract-errors which occur at a delivery-port.

8.3.1 *Abstract-operations*

This clause defines the following delivery-port abstract-operations:

a) message-delivery
b) report-delivery
c) delivery-control.

8.3.1.1 *Message-delivery*

The message-delivery abstract-operation enables the MTS to deliver a message to an MTS-user.

The MTS-user shall not refuse delivery of a message unless the delivery would violate the delivery-control restrictions then in force.

8.3.1.1.1 *Arguments*

Table 15/X.411 lists the arguments of the message-delivery abstract-operation, and for each argument qualifies its presence and identifies the clause in which the argument is defined.

8.3.1.1.1.1 *Message-delivery-identifier*

This argument contains an **MTS-identifer** that distinguishes the message from all other messages at the delivery-port. It shall be generated by the MTS, and shall have the same value as the **message-submission-identifier** supplied to the originator of the message when the message was submitted.

8.3.1.1.1.2 *Message-delivery-time*

This argument contains the **time** at which delivery occurs and at which the MTS is relinquishing responsibility for the message. It shall be generated by the MTS.

In the case of physical delivery, this argument indicates the **time** at which the PDAU has taken responsibility for printing and further delivery of the message.

The value of this argument shall be the same as the value of the **message-delivery-time** argument reported to the originator of the message (see § 8.3.1.2.1.8) in a delivery-report.

8.3.1.1.1.3 *This-recipient-name*

This argument contains the **OR-name** of the recipient to whom the message is being delivered. It shall be generated by the MTS.

The value of this argument shall be the same as the value of the **actual-recipient-name** argument reported to the originator of the message (see § 8.3.1.2.1.2) in a delivery-report.

The **this-recipient-name** contains the **OR-name** of the individual recipient, it shall not contain the **OR-name** of a DL.

The **OR-name** of the intended-recipient (if different, and the message has been redirected) is contained in the **intended-recipient-name** argument.

8.3.1.1.1.4 *Intended-recipient-name*

This argument contains the **OR-name** of the intended-recipient of the message if the message has been redirected and the time at which the redirection was performed. It may be generated by the MTS. A different value of this argument may be present for each occasion the message was redirected.

This argument comprises an **originally-intended-recipient-name** and an **intended-recipient-name**. On the first occasion a message is redirected, both the **originally-intended-recipient-name** and the **intended-recipient-name** contain the **recipient-name** originally-specified by the originator of the message. Subsequent redirections cause further **recipient-names** to be appended to the list of **intended-recipient-names**.

The **intended-recipient-name** contains the **OR-name** of an individual or DL intended-recipient and the time at which the message was redirected to an alternate recipient.

TABLE 15/X.411

Message-delivery arguments

Argument	Presence	Clause
Delivery arguments		
Message-delivery-identifier	M	8.3.1.1.1.1
Message-delivery-time	M	8.3.1.1.1.2
Message-submission-time	M	8.2.1.1.2.2
Originator argument		
Originator-name	M	8.2.1.1.1.1
Recipient arguments		
This-recipient-name	M	8.3.1.1.1.3
Intended-recipient-name	C	8.3.1.1.1.4
Redirection-reason	C	8.3.1.1.1.5
Other-recipient-names	C	8.3.1.1.1.6
DL-expansion-history	C	8.3.1.1.1.7
Priority argument		
Priority	C	8.2.1.1.1.8
Conversion arguments		
Implicit-conversion-prohibited	C	8.2.1.1.1.9
Conversion-with-loss-prohibited	C	8.2.1.1.1.10
Converted-encoded-information-types	C	8.3.1.1.1.8
Delivery method argument		
Requested-delivery-method	C	8.2.1.1.1.14
Physical delivery arguments		
Physical-forwarding-prohibited	C	8.2.1.1.1.15
Physical-forwarding-address-request	C	8.2.1.1.1.16
Physical-delivery-modes	C	8.2.1.1.1.17
Registered-mail-type	C	8.2.1.1.1.18
Recipient-number-for-advice	C	8.2.1.1.1.19
Physical-rendition-attributes	C	8.2.1.1.1.20
Originator-return-address	C	8.2.1.1.1.21
Physical-delivery-report-request	C	8.2.1.1.1.24
Security arguments		
Originator-certificate	C	8.2.1.1.1.25
Message-token	C	8.2.1.1.1.26
Content-confidentiality-algorithm-identifier	C	8.2.1.1.1.27
Content-integrity-check	C	8.2.1.1.1.28
Message-origin-authentication-check	C	8.2.1.1.1.29
Message-security-label	C	8.2.1.1.1.30
Proof-of-delivery-request	C	8.2.1.1.1.32
Content arguments		
Original-encoded-information-types	C	8.2.1.1.1.33
Content-type	M	8.2.1.1.1.34
Content-identifier	C	8.2.1.1.1.35
Content	M	8.2.1.1.1.37

8.3.1.1.1.5 *Redirection-reason*

This argument indicates the reason the message has been redirected to an alternate-recipient. It shall be generated by the MTS on each occasion that redirection occurs. A different value of this argument may be present for each occasion the message is redirected.

This argument may have one of the following values:

- **recipient-assigned-alternate-recipient**: the intended-recipient of the message requested that the message be redirected to a **recipient-assigned-alternate-recipient**; the originator of the message did not prohibit recipient-reassignment (see § 8.2.1.1.1.4); the MTS redirected the message to the **recipient-assigned-alternate-recipient**;
- **originator-requested-alternate-recipient**: the message could not be delivered to the intended-recipient or **recipient-assigned-alternate-recipient** (if registered); the **originator-requested-alternate-recipient** argument identified an alternate-recipient requested by the originator of the message; the MTS redirected the message to the **originator-requested-alternate-recipient**;
- **recipient-MD-assigned-alternate-recipient**: the **recipient-name** argument did not identify a recipient MTS-user; the **alternate-recipient-allowed** argument generated by the originator of the message allowed delivery to an alternate-recipient; the MTS redirected the message to an alternate-recipient assigned by the recipient-MD to receive such messages.

8.3.1.1.1.6 *Other-recipient-names*

This argument contains the originally-specified **OR-names** of all recipients other than those identified by the **originally-intended-recipient-name** argument, if present, and the **this-recipient-name** argument, if the originator of the message requested disclosure of other recipients (with the **disclosure-of-recipients** argument of the message-submission abstract-operation). It may be generated by the MTS. A different value of this argument may be present for each originally-specified recipient other than the **this-recipient-name** to which the message is being delivered.

Each **other-recipient-name** contains the **OR-name** of an individual recipient or a DL.

8.3.1.1.1.7 *DL-expansion-history*

This argument contains the sequence of **OR-names** of any DLs which have been expanded to add recipients to the copy of the message delivered to the recipient and the time of each expansion. It shall be generated by the MTS if any DL-expansion has occured.

8.3.1.1.1.8 *Converted-encoded-information-types*

This argument identifies the **encoded-information-types** of the message **content** after conversion, if conversion took place. It may be generated by the MTS.

8.3.1.1.2 *Results*

Table 16/X.411 lists the results of the message-delivery abstract-operation, and for each result qualifies its presence and identifies the clause in which the result is defined.

TABLE 16/X.411

Message-delivery results

Result	Presence	Clause
Proof of delivery results		
Recipient-certificate	O	8.3.1.1.2.1
Proof-of-delivery	C	8.3.1.1.2.2

8.3.1.1.2.1 *Recipient-certificate*

This argument contains the **certificate** of the recipient of the message. It shall be generated by a trusted source (e.g. certification-authority), and may be supplied by the recipient of the message, if the originator of the message requested **proof-of-delivery** (see § 8.2.1.1.1.32) and an asymmetric-encryption-algorithm is used to compute the **proof-of-delivery**.

The **recipient-certificate** may be used to convey a verified copy of the public-asymmetric-encryption-key (**subject-public-key**) of the recipient of the message.

The recipient's public-asymmetric-encryption-key may be used by the originator of the message to validate the **proof-of-delivery**.

8.3.1.1.2.2 *Proof-of-delivery*

This argument provides the originator of the message with proof that the message has been delivered to the recipient (to provide the proof of delivery element-of-service as defined in Recommendation X.400). Depending on the encryption-algorithm used and the security-policy in force, this argument may also provide the non-repudiation of delivery element-of-service (as defined in Recommendation X.400). It shall be generated by the recipient of the message, if the originator of the message requested **proof-of-delivery** (see § 8.2.1.1.1.32).

The **proof-of-delivery** is computed using the algorith identified by the **proof-of-delivery-algorithm-identifier** (an **algorithm-identifier**).

The **proof-of-delivery** contains the **proof-of-delivery-algorithm-identifier**, and an encrypted function (e.g., a compressed or hashed version) of the **proof-of-delivery-algorithm-identifier**, the **delivery-time**, and the **this-recipient-name**, the **originally-intended-recipient-name**, the message **content**, the **content-identifier**, and the **message-security-label** of the delivered message. Optional components are included in the **proof-of-delivery** if they are present in the delivered message. Note that the **proof-of-delivery** is computed using the clear (i.e. unencrypted) message **content**.

Note that receipt of this argument provides the originator of the message with proof of delivery of the message to the recipient. Non-receipt of this argument provides neither proof of delivery nor proof of non-delivery (unless a secure route and trusted functionality are employed).

If an asymmetric-encryption-algorithm is used, the **proof-of-delivery** may be computed by the recipient of the message using the recipient's secret-asymmetric-encryption-key. The originator of the message may validate the **proof-of-delivery** using the recipient's public asymmetric-encryption-key (**subject-public-key**) derived from the **recipient-certificate**. An asymmetric **proof-of-delivery** may also provide for non-repudiation of delivery.

If a symmetric-algorithm is used, a symmetric-encryption-key is used by the recipient to compute the **proof-of-delivery**, and by the originator to validate the **proof-of-delivery**. Note that if a symmetric-encryption-algorithm is used then the **proof-of-delivery** can only provide non repudiation of delivery if the security-policy in force provides for the involvement of a third party acting as a notary. The means by which the symmetric-encryption-key is distributed is not currently defined by this Recommendation.

8.3.1.1.3 *Abstract-errors*

Table 17/X.411 lists the abstract-errors that may disrupt the message-delivery abstract-operation, and for each abstract-error identifies the clause in which the abstract-error is defined.

TABLE 17/X.411

Message-delivery abstract-errors

Abstract-error	Clause
Delivery-control-violated	8.3.2.1
Security-error	8.3.2.3
Unsupported-critical-function	8.3.2.4

8.3.1.2 *Report-delivery*

The **report-delivery** abstract-operation enables the MTS to acknowledge to the MTS-user one or more outcomes of a previous invocation of the message-submission or probe-submission abstract-operations.

For the message-submission abstract-operation, the report-delivery abstract-operation indicates the delivery or non-delivery of the submitted message to one or more recipients.

For the probe-submission abstract-operation, the report-delivery abstract-operation indicates whether or not a message could be delivered, or a DL-expansion could occur, if the message were to be submitted.

A single invocation of the message-submission or probe-submission abstract-operation may provoke several occurences of the report-delivery abstract-operation, each covering one or more intended recipients. A single occurence of the report-delivery abstract-operation may report on both delivery and non-delivery to different recipients.

An invocation of the message-submission or probe-submission abstract-operation by one MTS-user may provoke occurences of the report-delivery abstract-operation to another MTS-user, i.e., reports delivered to the owner of a DL.

The MTS-user shall not refuse to accept the delivery of a report unless the delivery of the report would violate the delivery-control restrictions then in force.

8.3.1.2.1 *Arguments*

Table 18/X.411 lists the arguments of the report-delivery abstract-operation, and for each argument qualifies its presence and identifies the clause in which the argument is defined.

8.3.1.2.1.1 *Subject-submission-identifier*

This argument contains the **message-submission-identifer** or the **probe-submission-identifier** of the subject of the report. It shall be supplied by the MTS.

8.3.1.2.1.2 *Actual-recipient-name*

This argument contains the **OR-name** of a recipient of the message. It shall be generated by the originator of the message, or by the MTS if the message has been redirected. A different value of this argument shall be specified for each recipient of the subject to which this report relates.

In the case of a delivery report, the **actual-recipient-name** is the name of the actual recipient of the message, and has the same value as the **this-recipient-name** argument of the delivered message. In the case of a non-delivery-report, the **actual-recipient-name** is the **OR-name** of the recipient to which the message was being directed when the reason for non-delivery was encountered.

The **actual-recipient-name** may be an originally-specified **recipient-name**, or the **OR-name** of an alternate-recipient if the message has been redirected. If the message has been redirected, the **OR-name** of the intended-recipient is contained in the **intended-recipient-name** argument.

The **actual-recipient-name** contains the **OR-name** of an individual recipient or DL.

8.3.1.2.1.3 *Originator-and-DL-expansion-history*

This argument contains a sequence of **OR-names** and associated times which document the history of the origin of the subject-message. This first **OR-name** in the sequence is the **OR-name** of the originator of the subject, and the remainder of the sequence is a sequence of **OR-names** of the DLs that have been expanded in directing the subject towards the recipient (the latter being the same as the **DL-expansion-history**). It shall be generated by the originating-MTA of the report if any DL-expansion has occurred on the subject.

The **originator-and-DL-expansion-history** contains the **OR-name** of the originator of the subject and each DL and the time at which the associated event occurred.

8.3.1.2.1.4 *Reporting-DL-name*

This argument contains the **OR-name** of the DL that forwarded the report to the owner of the DL. It shall be generated by a DL-expansion-point (an MTA) when forwarding a report to the owner of the DL, in line with the reporting-policy of the DL.

The **reporting-DL-name** contains the **OR-name** of the DL forwarding the report.

TABLE 18/X.411

Report-delivery arguments

Argument	Presence	Clause
Subject submission argument		
Subject-submission-identifier	M	8.3.1.2.1.1
Recipient arguments		
Actual-recipient-name	M	8.3.1.2.1.2
Intended-recipient-name	C	8.3.1.1.1.4
Redirection-reason	C	8.3.1.1.1.5
Originator-and-DL-expansion-history	C	8.3.1.2.1.3
Reporting-DL-name	C	8.3.1.2.1.4
Conversion arguments		
Converted-encoded-information-types	C	8.3.1.2.1.5
Supplementary information arguments		
Supplementary-information	C	8.3.1.2.1.6
Physical-forwarding-address	C	8.3.1.2.1.7
Delivery arguments		
Message-delivery-time	C	8.3.1.2.1.8
Type-of-MTS-user	C	8.3.1.2.1.9
Non-delivery arguments		
Non-delivery-reason-code	C	8.3.1.2.1.10
Non-delivery-diagnostic-code	C	8.3.1.2.1.11
Security arguments		
Recipient-certificate	C	8.3.1.1.2.1
Proof-of-delivery	C	8.3.1.1.2.2
Reporting-MTA-certificate	C	8.3.1.2.1.12
Report-origin-authentication-check	C	8.3.1.2.1.13
Message-security-label	C	8.2.1.1.1.30
Content arguments		
Original-encoded-information-types	C	8.2.1.1.1.33
Content-type	C	8.2.1.1.1.34
Content-identifier	C	8.2.1.1.1.35
Content-correlator	C	8.2.1.1.1.36
Returned-content	C	8.3.1.2.1.14

8.3.1.2.1.5 *Converted-encoded-information-types*

This argument identifies the **encoded-information-types** of the subject-message **content** after conversion, if conversion took place. For a report on a message, this argument indicates the actual **encoded-information-types** of the converted message **content**. For a report on a probe, this argument indicates the **encoded-information-types** the subject-message **content** would have contained after conversion, if the subject-message were to have been submitted. It may be generated by the MTS. A different value of this parameter may be specified for each recipient of the subject to which the report relates.

8.3.1.2.1.6 *Supplementary-information*

This argument may contain information supplied by the originator of the report, as a printable string. It may be generated by the originating-MTA of the report or an associated access-unit. A different value of this argument may be specified for each intended recipient of the subject to which the report relates.

Supplementary-information may be used by a Teletex-access-unit or a Teletex/Telex conversion facility. It may contain a received answer-back, Telex transmission duration, or note and received recorded message as a printable string.

Supplementary-information may also be used by other access-units, or by the originating-MTA of the report itself, to convey printable information to the originator of the message.

8.3.1.2.1.7 *Physical-forwarding-address*

This argument contains the new **postal-OR-address** of the physical-recipient of the message. It may be generated by the associated PDAU of the originating-MTA of the report, if the originator of the message requested the physical-forwarding-address of the recipient (see § 8.2.1.1.1.16). A different value of this argument may be specified for each intended recipient of the subject-message to which the report relates.

8.3.1.2.1.8 *Message-delivery-time*

This argument contains the **time** at which the subject-message was (or would have been) delivered to the recipient MTS-user. It shall be generated by the MTS if the message was (or would have been) successfully delivered. A different value of this argument may be specified for each intended-recipient of the subject to which the report relates.

In the case of physical delivery, this argument indicates the **time** at which the PDAU has taken responsibility for printing and further delivery of the message.

If the subject-message was delivered, the value of this argument shall be the same as the value of the **message-delivery-time** argument of the delivered message (see § 8.3.1.1.1.2).

8.3.1.2.1.9 *Type-of-MTS-user*

This argument indicates the type of recipient MTS-user to which the message was (or would have been) delivered. It shall be generated by the MTS if the message was (or would have been) successfully delivered. A different value of this argument may be specified for each intended-recipient of the subject to which the report relates.

This argument may have one of the following values:

- **public**: a UA owned by an Administration;
- **private**: a UA owned by other than an Administration;
- **ms**: a message-store;
- **DL**: a distribution-list;
- **PDAU**: a physical-delivery-access-unit (PDAU);
- **physical-recipient**: a physical-recipient of a PDS;
- **other**: an access-unit of another kind.

8.3.1.2.1.10 *Non-delivery-reason-code*

This argument contains a code indicating the reason the delivery of the subject-message failed (or, in the case of a probe, would have failed). It shall be generated by the MTS if the message was (or would have been) unsuccessfully delivered. A different value of this argument may be specified for each intended-recipient of the subject to which the report relates.

This argument may have one of the following values:

- **transfer-failure**: indicates that, while the MTS was attempting to deliver or probe delivery of the subject-message, some communication failure prevented it from doing so;
- **unable-to-transfer**: indicates that, due to some problem with the subject itself, the MTS could not deliver or probe delivery of the subject-message;
- **conversion-not-performed**: indicates that a conversion necessary for the delivery of the subject-message was (or would be) unable to be performed;
- **physical-rendition-not-performed**: indicates that the PDAU was unable to physically render the subject-message;

- **physical-delivery-not-performed**: indicates that the PDS was unable to physically deliver the subject-message;
- **restricted-delivery**: indicates that the recipient subscribes to the restricted-delivery element-of-service (as defined in Recommendation X.400) which prevented (or would prevent) the delivery of the subject-message;
- **directory-operation-unsuccessful**: indicates that the outcome of a required directory operation was unsuccessful.

Other **non-delivery-reason-codes** may be specified in future versions of this Recommendation.

Further information on the nature of the problem preventing delivery is contained in the **non-delivery-diagnostic-code** argument.

8.3.1.2.1.11 *Non-delivery-diagnostic-code*

This argument contains a code indicating the nature of the problem which caused delivery or probing of delivery of the subject-message to fail. The reason for failure is indicated by the **non-delivery-reason-code** argument. It may be generated by the MTS if the message was (or would have been) unsuccessfully delivered. A different value of this argument may be specified for each intended-recipient of the subject to which the report relates.

This argument may have one of the following values:

- **unrecognised-OR-name**: the **recipient-name** argument of the subject does not contain an **OR-name** recognised by the MTS;
- **ambiguous-OR-name**: the **recipient-name** argument of the subject identifies more than one potential recipient (i.e., is ambiguous);
- **MTS-congestion**: the subject could not be progressed, due to congestion in the MTS;
- **loop-detected**: the subject was detected looping within the MTS;
- **recipient-unavailable**: the recipient MTS-user was (or would be) unavailable to take delivery of the subject-message;
- **maximum-time-expired**: the maximum time for delivering the subject-message, or performing the subject-probe, expired;
- **encoded-information-types-unsupported**: the **encoded-information-types** of the subject-message are unsupported by the recipient MTS-user;
- **content-too-long**: the **content-length** of the subject-message is too long for the recipient MTS-user to take delivery (exceeds the **deliverable-maximum-content-length**);
- **conversion-impractical**: a conversion required for the subject-message to be delivered is impractical;
- **implicit-conversion-prohibited**: a conversion required for the subject-message to be delivered has been prohibited by the originator of the subject (see § 8.2.1.1.1.9);
- **implicit-conversion-not-subscribed**: a conversion required for the subject-message to be delivered has not been subscribed to by the recipient;
- **invalid-arguments**: one or more arguments in the subject was detected as being invalid;
- **content-syntax-error**: a syntax error was detected in the **content** of the subject-message (not applicable to subject-probes);
- **size-constraint-violation**:indicates that the value of one or more parameter(s) of the subject violated the size constraints defined in this Recommendation, and that the MTS was not prepared to handle the specified value(s);
- **protocol-violation**: indicates that one or more mandatory argument(s) were missing from the subject;
- **content-type-not-supported**: indicates that processing of a **content-type** not supported by the MTS was (or would be) required to deliver the subject-message;
- **too-many-recipients**: indicates that the MTS was (or would be) unable to deliver the subject-message due to the number of specified recipients of the subject-message (see § 8.2.1.1.1.2);
- **no-bilateral-agreement**: indicates that delivery of the subject-message required (or would require) a bilateral agreement where no such agreement exists;

- **unsupported-critical-function**: indicates that a critical function required for the transfer or delivery of the subject-message was not suported by the originating-MTA of the report;
- **conversion-with-loss-prohibited**: a conversion required for the subject-message to be delivered would have resulted in loss of information; conversion with loss of information was prohibited by the originator of the subject (see § 8.2.1.1.1.10);
- **line-too-long**: a conversion required for the subject-message to be delivered would have resulted in loss of information because the original line length was too long;
- **page-split**: a conversion required for the subject-message to be delivered would have resulted in loss of information because an original page would be split;
- **pictorial-symbol-loss**: a conversion required for the subject-message to be delivered would have resulted in loss of information because of a loss of one or more pictorial symbols;
- **punctuation-symbol-loss**: a conversion required for the subject-message to be delivered would have resulted in loss of information because of a loss of one or more punctuation symbols;
- **alphabetic-character-loss**: a conversion required for the subject-message to be delivered would have resulted in loss of information because of a loss of one or more alphabetic characters;
- **multiple-information-loss**: a conversion required for the subject-message to be delivered would have resulted in multiple loss of information;
- **recipient-reassignment-prohibited**: indicates that the MTS was (or would be) unable to deliver the subject-message because the originator of the subject prohibited redirection to a **recipient-assigned-alternate-recipient** (see § 8.2.1.1.1.4);
- **redirection-loop-detected**: the subject-message could not be redirected to an alternate-recipient because that recipient had previously redirected the message (redirection-loop);
- **DL-expansion-prohibited**: indicates that the MTS was (or would be) unable to deliver the subject-message because the originator of the subject prohibited the expansion of DLs (see § 8.2.1.1.1.6);
- **no-DL-submit-permission**: the originator of the subject (or the DL of which this DL is a member, in the case of nested DLS) does not have permission to submit messages to this DL;
- **DL-expansion-failure**: indicates that the MTS was unable to complete the expansion of a DL;
- **physical-rendition-attributes-not-supported**: the PDAU does not support the physical-rendition-attributes requested (see § 8.2.1.1.1.20);
- **undeliverable-mail-physical-delivery-address-incorrect**: the subject-message was undeliverable because the specified recipient **postal-OR-address** was incorrect;
- **undeliverable-mail-physical-delivery-office-incorrect-or-invalid**: the subject-message was undeliverable because the physical-delivery-office identified by the specified recipient **postal-OR-address** was incorrect or invalid (does not exit);
- **undeliverable-mail-physical-delivery-address-incomplete**: the subject-message was undeliverable because the specified recipient **postal-OR-address** was incompletely specified;
- **undeliverable-mail-recipient-unknown**: the subject-message was undeliverable because the recipient specified in the recipient **postal-OR-address** was not known at that address;
- **undeliverable-mail-recipient-deceased**: the subject-message was undeliverable because the recipient specified in the recipient **postal-OR-address** is deceased;
- **undeliverable-mail-organization-expired**: the subject-message was undeliverable because the recipient organization specified in the recipient **postal-OR-address** has expired;
- **undeliverable-mail-recipient-refused-to-accept**: the subject-message was undeliverable because the recipient specified in the recipient **postal-OR-address** refused to accept it;
- **undeliverable-mail-recipient-did-not-claim**: the subject-message was undeliverable because the recipient specified in the recipient **postal-OR-address** did not collect the mail;
- **undeliverable-mail-recipient-changed-address-permanently**: the subject-message was undeliverable because the recipient specified in the recipient **postal-OR-address** has changed address permanently ('moved'), and forwarding was not applicable;
- **undeliverable-mail-recipient-changed-address-temporarily**: the subject-message was undeliverable because the recipient specified in the recipient **postal-OR-address** has changed address temporarily ('on travel'), and forwarding was not applicable;
- **undeliverable-mail-recipient-changed-temporary-address**: the subject-message was undeliverable because the recipient specified in the recipient **postal-OR-address** had changed temporary address ('departed'), and forwarding was not applicable;
- **undeliverable-mail-new-address-unknown**: the subject-message was undeliverable because the recipient has moved and the recipient's new address is unknown;
- **undeliverable-mail-recipient-did-not-want-forwarding**: the subject-message was undeliverable because delivery would have required physical-forwarding which the recipient did not want;

- **undeliverable-mail-originator-prohibited-forwarding**: the physical-forwarding required for the subject-message to be delivered has been prohibited by the originator of the subject-message (see § 8.2.1.1.1.15);
- **secure-messaging-error**: the subject could not be progressed because it would violate the security-policy in force;
- **unable-to-downgrade**: the subject could not be transferred because it could not be downgraded (see Annex B to Recommendation X.419).

Other **non-delivery-diagnostic-codes** may be specified in future versions of this Recommendation.

8.3.1.2.1.12 *Reporting-MTA-certificate*

This argument contains the **certificate** of the MTA that generated the report. It shall be generated by a trusted source (e.g., a certification-authority), and may be supplied by the reporting-MTA if a **report-origin-authentication-check** is supplied.

The **reporting-MTA-certificate** may be used to convey a verified copy of the public-asymmetric-encription-key (**subject-public-key**) of the reporting-MTA.

The reporting-MTA's public-asymmetric-encryption-key may be used by the originator of the message, and any MTA through which the report is transferred, to validate the **report-origin-authentication-check**.

8.3.1.2.1.13 *Report-origin-authentication-check*

This argument provides the originator of the subject-message (or -probe), and any other MTA through which the report is transferred, with a means of authenticating the origin of the report (to provide the report origin authentication element-of-service as defined in Recommendation X.400). It may be generated by the reporting-MTA if a **message-** (or **probe-**) **origin-authentication-check** was present in the subject.

The **report-origin-authentication-check** provides proof of the origin of the report (report origin authentication), and proof of association between the **message-security-label** and the report.

The **report-origin-authentication-check** is computed using the algorithm identified by the **report-origin-authentication-algorithm-identifier** (an **algorithm-identifier**).

The **report-origin-authentication-check** contains the **report-origin-authentication-algorithm-identifier**, and an asymmetrically encrypted, hashed version of the **report-origin-authentication-algorithm-identifier**, the **content-identifier** and **message-security-label** of the subject, and all values of the following (per-recipient) arguments: the **actual-recipient-name**, the **originally-intended-recipient-name**, and:

- for a delivery-report: the **message-delivery-time**, the **type-of-MTS-user**, and if requested by the originator of the message for recipients to which the report relates, the **recipient-certificate**, and the **proof-of-delivery** (not present in a report on a probe); or
- for a non-delivery-report: the **non-delivery-reason-code** and **non-delivery-diagnostic-code**.

Optional components are included in the **report-authentication-check** if they are present in the report.

The **report-origin-authentication-check** may be computed by the reporting-MTA using the reporting-MTA's secret-asymmetric-encryption-key. The **report-origin-authentication-check** may be validated by the originator of the subject, and any MTA through which the report is transferred, using the reporting-MTA's public-asymmetric-encryption-key (**subject-public-key**) derived from the **reporting-MTA-certificate**.

Future versions of this Recommendation may define other forms of **report-origin-authentication-check** (e.g., based on symmetric-encryption-techniques) which may be used by MTAs through which the report is transferred to authenticate the origin of the report.

8.3.1.2.1.14 *Returned-content*

This argument contains the **content** of the subject-message if the originator of the subject-message indicated that the **content** was to be returned (see § 8.2.1.1.1.23). It shall be generated by the originator of the message, and may be returned by the MTS (if the reporting-MTA or originating-MTA supports the return of content element-of-service).

This argument may only be present if there is at least one non-delivery report in the Report-delivery, and if the recipient of the report is the originator of the subject-message (and not, for example, the owner of a DL (see § 8.3.1.2.1.4)).

This argument shall not be present if any **encoded-information-type** conversion has been performed on the **content** of the subject-message.

8.3.1.2.2 *Results*

The report-delivery abstract-operation returns an empty result as indication of success.

8.3.1.2.3 *Abstract-errors*

Table 19/X.411 lists the abstract-errors that may disrupt the report-delivery abstract-operation, and for each abstract-error identifies the clause in which the abstract-error is defined.

TABLE 19/X.411

Report-delivery abstract-errors

Abstract-error	Clause
Delivery-control-violated	8.3.2.1
Security-error	8.3.2.3
Unsupported-critical-function	8.3.2.4

8.3.1.3 *Delivery-control*

The delivery-control abstract-operation enables the MTS-user to temporarily limit the delivery-port abstract-operations that the MTS may invoke, and the messages that the MTS may deliver to the MTS-user via the message-delivery abstract-operation.

The MTS shall hold until a later time, rather than abandon, abstract-operations and messages presently forbidden.

The successful completion of the abstract-operation signifies that the specified controls are now in force. These controls supersede any previously in force, and remain in effect until the association is released, the MTS-user re-invokes the delivery-control abstract-operation, or the MTS-user invokes the administration-port register abstract-operation to impose constraints more severe than the specified controls.

The abstract-operation returns an indication of any abstract-operations that the MTS would invoke, or any message types that the MTS would deliver or report, were it not for the prevailing controls.

8.3.1.3.1 *Arguments*

Table 20/X.411 lists the arguments of the delivery-control abstract-operation, and for each argument qualifies its presence and identifies the clause in which the argument is defined.

8.3.1.3.1.1 *Restrict*

This argument indicates whether the controls on delivery-port abstract-operations are to be updated or removed. It may be generated by the MTS-user.

This argument may have one of the following values:

- **update**: the other arguments update the prevailing controls;
- **remove**: all temporary controls are to be removed (the default controls registered with the MTS by means of the administration-port register abstract-operation shall apply); the other arguments are to be ignored.

In the absence of this argument, the default **update** shall be assumed.

TABLE 20/X.411

Delivery-control arguments

Arguments	Presence	Clause
Delivery control arguments		
Restrict	O	8.3.1.3.1.1
Permissible-operations	O	8.3.1.3.1.2
Permissible-lowest-priority	O	8.3.1.3.1.3
Permissible-encoded-information-types	O	8.3.1.3.1.4
Permissible-content-types	O	8.3.1.3.1.5
Permissible-maximum-content-length	O	8.3.1.3.1.6
Permissible-security-context	O	8.3.1.3.1.7

8.3.1.3.1.2 *Permissible-operations*

This argument indicates the abstract-operations that the MTS may invoke on the MTS-user. It may be generated by the MTS-user.

This argument may have the value **allowed** or **prohibited** for each of the following:

- **message-delivery**: the MTS may/may not invoke the message-delivery abstract-operation; and
- **report-delivery**: the MTS may/may not invoke the report-delivery abstract-operation.

Other delivery-port abstract-operations are not subject to controls, and may be invoked at any time.

In the absence of this argument, the abstract-operations that the MTS may invoke on the MTS-user are unchanged. If there has been no previous invocation of the delivery-control abstract-operation on the association, the default control registerd with the MTS by means of the administration-port Register abstract-operation shall apply.

8.3.1.3.1.3 *Permissible-lowest-priority*

This argument contains the **priority** of the lowest priority message that the MTS shall deliver to the MTS-user via the message-delivery abstract-operation. It may be generated by the MTS-user.

This argument may have one of the following values of the **priority** argument of the message-submission abstract-operation: **normal, non-urgent** or **urgent.**

In the absence of this argument, the **priority** of the lowest priority message that the MTS shall deliver to the MTS-user is unchanged. If there has been no previous invocation of the delivery-control abstract-operation on the association, the default control registered with the MTS by means of the adminsitration-port Register abstract-operation shall apply.

8.3.1.3.1.4 *Permissible-encoded-information-types*

This argument indicates the only **encoded-information-types** that shall appear in messages that the MTS shall deliver to the MTS-user via the message-delivery abstract-operation. It may be generated by the MTS-user.

The **permissible-encoded-information-types** specified shall be among those allowed long-term due to a previous invocation of the administration-port register abstract-operation (**deliverable-encoded-information-types**).

In the absence of this argument, the **permissible-encoded-information-types** that the MTS may deliver to the MTS-user are unchanged. If there has been no previous invocation of the delivery-control abstract-operation on the association, the default control registered with the MTS by means of the administration-port register abstract-operation shall apply.

8.3.1.3.1.5 *Permissible-content-types*

This argument indicates the only **content-types** that shall appear in messages that the MTS shall deliver to the MTS-user via the message-delivery abstract-operation. It may be generated by the MTS-user.

The **permissible-content-types** specified shall be among those allowed long-term due to a previous invocation of the administration-port register abstract-operation (**deliverable-content-types**).

In the absence of this argument, the **permissible-content-types** that the MTS may deliver to the MTS-user are unchanged. If there has been no previous invocation of the delivery-control abstract-operation on the association, the default control registered with the MTS by means of the administration-port register abstract-operation shall apply.

8.3.1.3.1.6 *Permissible-maximum-content-length*

This argument contains the **content-length**, in octets, of the longest-content message that the MTS shall deliver to the MTS-user via the message-delivery abstract-operation. It may be generated by the MTS-user.

The **persmissible-maximum-content-length** shall not exceed that allowed long-term due to a previous invocation of the administration-port register abstract-operation (**deliverable-maximum-content-length**).

In the absence of this argument, the **permissible-maximum-content-length** of a message that the MTS may deliver to the MTS-user is unchanged. If there has been no previous invocation of the delivery-control abstract-operation on the association, the default control registered with the MTS by means of the administration port register abstract-operation shall apply.

8.3.1.3.1.7 *Permissible-security-context*

This argument temporarily limits the sensitivity of delivery-port abstract-operations (delivery-security-context) that the MTS may invoke on the MTS-user. It is a temporary restriction of the **security-context** established when the association was initiated (see § 8.1.1.1.1.4). It may be generated by the MTS-user.

The **permissible-security-context** comprises one or more **security-labels** from the set of **security-labels** established as the **security-context** when the association was established.

In the absence of this argument, the **security-context** of delivery-port abstract-operations is unchanged.

8.3.1.3.2 *Results*

Table 21/X.411 lists the results of the delivery-control abstract-operation, and for each result qualifies its presence and identifies the clause in which the result is defined.

TABLE 21/X.411

Delivery-control results

Results	Presence	Clause
"Waiting" results		
Waiting-operations	O	8.3.1.3.2.1
Waiting-messages	O	8.3.1.3.2.2
Waiting-encoded-information-types	O	8.3.1.3.2.3
Waiting-content-types	O	8.3.1.3.2.4

8.3.1.3.2.1 *Waiting-operations*

This result indicates the abstract-operations being held by the MTS,and that the MTS would invoke on the MTS-user if it were not for the prevailing controls. It may be generated by the MTS.

This result may have the value **holding** or **not-holding** for each of the following:

- **message-delivery**: the MTS is/is not holding messages, and would invoke the message-delivery abstract-operation on the MTS-user if it were not for the prevailing controls; and
- **report-delivery**: the MTS is/is not holding reports, and would invoke the report-delivery abstract-operation on the MTS-user if it were not for the prevailing controls.

In the absence of this result, it may be assumed that the MTS is not holding any messages or reports for delivery due to the prevailing controls.

8.3.1.3.2.2 *Waiting-messages*

This result indicates the kind of messages the MTS is holding for delivery to the MTS-user, and would deliver via the message-delivery abstract-operation, if it were not for the prevailing controls. It may be generated by the MTS.

This result may have one or more of the following values:

- **long-content**: the MTS has messages held for delivery to the MTS-user which exceed the **permissible-maximum-content-length** control currently in force;
- **low-priority**: the MTS has messages held for delivery to the MTS-user of a lower priority than the **permissible-lowest-priority** control currently in force;
- **other-security-labels**: the MTS has messages held for delivery to the MTS-user bearing **message-security-labels** other than those permitted by the current security-context.

In the absence of this result, it may be assumed that the MTS is not holding any messages for delivery to the MTS-user due to the **permissible-maximum-content-length**, **permissible-lowest-priority** or **permissible-security-context** controls currently in force.

8.3.1.3.2.3 *Waiting-encoded-information-types*

This result indicates the **encoded-information-types** in the **content** of any messages held by the MTS for delivery to the MTS-user due to prevailing controls. It may be generated by the MTS.

In the absence of this result, the **encoded-information-types** of any messages held by the MTS for delivery to the MTS-user are **unspecified**.

8.3.1.3.2.4 *Waiting-content-types*

This result indicates the **content-types** of any messages held by the MTS for delivery to the MTS-user due to prevailing controls. It may be generated by the MTS.

In the absence of this result, the **content-types** of any messages held by the MTS for delivery to the MTS-user are **unspecified**.

8.3.1.3.3 *Abstract-errors*

Table 22/X.411 lists the abstract-errors that may disrupt the delivery-control abstract-operation, and for each abstract-error identifies the clause in which the abstract-error is defined.

TABLE 22/X.411

Delivery-control abstract-errors

Abstract-error	Clause
Control-violates-registration	8.3.2.2
Security-error	8.3.2.3

8.3.2 *Abstract-errors*

This clause defines the following delivery-port abstract-errors:

a) delivery-control-violated
b) control-violates-registration
c) security-error
d) unsupported-critical-function.

8.3.2.1 *Delivery-control-violated*

The delivery-control-violated abstract-error reports the violation by the MTS of a control on delivery-port abstract-operations imposed by the MTS-user via the delivery-control abstract-operation.

The delivery-control-violated abstract-error has no parameters.

8.3.2.2 *Control-violates-registration*

The control-violates-registration abstract-error reports that the MTS is unable to accept the controls that the MTS-user attempted to impose on delivery-port abstract-operations because they violate existing registration parameters.

The control-violates-registration abstract-error has no parameters.

8.3.2.3 *Security-error*

The security-error abstract-error reports that the requested abstract-operation could not be provided by the MTS-user because it would violate the security-policy in force.

The security-error abstract-error has the following parameters, generated by the MTS-user:

– **security-problem**: an identifier for the cause of the violation of the security-policy.

8.3.2.4 *Unsupported-critical-function*

The unsupported-critical-function abstract-error reports that an argument of the abstract-operation was marked **critical-for-delivery** (see § 9.1) but is unsupported by the MTS-user.

The unsupported-critical-function abstract-error has no parameters.

8.4 *Administration port*

This section defines the abstract-operations and abstract-errors which occur at an administration-port.

8.4.1 *Abstract-operations*

This section defines the following administration-port abstract-operations:

a) register
b) change-credentials.

8.4.1.1 *Register*

The register abstract-operation enables an MTS-user to make long-term changes to various parameters of the MTS-user held by the MTS concerned with delivery of messages to the MTS-user.

Such changes remain in effect until overridden by re-invocation of the register abstract-operation. However, some parameters may be temporarily overridden by invocation of the delivery-control abstract-operation.

Note 1 – This abstract-operation shall be invoked before any other submission-port, delivery-port or administration-port abstract-operation may be used, or an equivalent registration by local means shall have taken place.

Note 2 – This abstract-operation does not encompass the standing parameters implied by the alternate recipient allowed element-of-service and the restricted delivery element-of-service defined in Recommendation X.400. The manner in which those parameters are supplied and modified are a local matter.

8.4.1.1.1 *Arguments*

Table 23/X.411 lists the arguments of the register abstract-operation, and for each argument qualifies its presence and identifies the section in which the argument is defined.

TABLE 23/X.411

Register arguments

Argument	Presence	Clause
Registration arguments		
User-name	O	8.4.1.1.1.1
User-address	O	8.4.1.1.1.2
Deliverable-encoded-information-types	O	8.4.1.1.1.3
Deliverable-content-types	O	8.4.1.1.1.4
Deliverable-maximum-content-length	O	8.4.1.1.1.5
Recipient-assigned-alternate-recipient	O	8.4.1.1.1.6
User-security-labels	O	8.4.1.1.1.7
Default delivery control arguments		8.4.1.1.1.8
Restrict	O	8.3.1.3.1.1
Permissible-operations	O	8.3.1.3.1.2
Permissible-lowest-priority	O	8.3.1.3.1.3
Permissible-encoded-information-types	O	8.3.1.3.1.4
Permissible-content-types	O	8.3.1.3.1.5
Permissible-maximum-content-length	O	8.3.1.3.1.6

8.4.1.1.1.1 *User-name*

This argument contains the **OR-name** of the MTS-user, if the **user-name** is to be changed. It may be generated by the MTS-user.

In the absence of this argument, the **user-name** of the MTS-user remains unchanged.

An MD is not required to provide MTS-users with the ability to change their **OR-names**. If it does so, the MD may restrict that ability. It may prohibit certain MTS-users from changing their **OR-names**, or it may restrict the scope of the change to a locally defined subset of the components of their **OR-names**. A proposed new **OR-names** shall be rejected if it is already assigned to another MTS-user.

8.4.1.1.1.2 *User-address*

This argument contains the **user-address** of the MTS-user, if it is required by the MTS and if it is to be changed. It may be generated by the MTS-user.

The user-address may contain one of the following forms of address of the MTS-user.

- the **X.121-address** and/or the **TSAP-ID** (transport service access point identifier); or
- the **PSAP-address** (presentation service access point address).

Other forms of **user-address** may be defined in future versions of this Recommendation.

In the absence of this argument, the **user-address** of the MTS-user (if any) remains unchanged.

8.4.1.1.1.3 *Deliverable-encoded-information-types*

This argument indicates the **encoded-information-types** that the MTS shall permit to appear in messages delivered to the MTS-user, if they are to be changed. It may be generated by the MTS-user.

The MTS shall reject as undeliverable any message for an MTS-user for which the MTS-user is not registered to accept delivery of all the **encoded-information-types** of the message. Note that the MTS-user may register to receive the **undefined encoded-information-type**. Deliverable-encoded-information-types also indicates the possible encoded-information-types to which implicit conversion can be performed.

In the absence of this argument, the **deliverable-encoded-information-types** shall remain unchanged.

8.4.1.1.1.4 *Deliverable-content-types*

This argument indicates the **content-types** that the MTS shall permit to appear in messages delivered to the MTS-user, if they are to be changed. It may be generated by the MTS-user.

The MTS shall reject as undeliverable any message for an MTS-user for which the MTS-user is not registered to accept delivery of the **content-types** of the message. Note that the MTS-user may register to receive the **undefined content-type**.

In the absence of this argument, the **deliverable-content-types** shall remain unchanged.

8.4.1.1.1.5 *Deliverable-maximum-content-length*

This argument contains the **content-length**, in octets, of the longest-content message that the MTS shall permit to appear in messages delivered to the MTS-user, if it is to be changed. It may be generated by the MTS-user.

The MTS shall reject as undeliverable any message for an MTS-user for which the MTS-user is not registered to accept delivery of messages of its size.

In the absence of this argument, the **deliverable-maximum-content-length** of messages shall remain unchanged.

8.4.1.1.1.6 *Recipient-assigned-alternate-recipient*

This argument contains the **OR-name** of an alternate-recipient, specified by the MTS-user, to which messages are to be redirected, if the alternate-recipient is to be changed. It may be generated by the MTS-user. A different value of this argument may be specified for each value of **user-security-labels**.

If a **recipient-assigned-alternate-recipient** is registered and associated with a value of **user-security-labels**, messages bearing a matching **message-security-label** shall be redirected to the alternate-recipient. Messages bearing a **message-security-label** for which no **recipient-assigned-alternate-recipient** has been registered, shall not be redirected to a **recipient-assigned-alternate-recipient**.

If a single **recipient-assigned-alternate-recipient** is registered, and not associated with a value of **user-security-labels**, all messages shall be redirected to the alternate-recipient.

The **recipient-assigned-alternate-recipient** shall contain the **OR-name** of the alternate-recipient. If the **recipient-assigned-alternate-recipient** contains the **OR-names** of the MTS-user (see § 8.4.1.1.1.1), no **recipient-assigned-alternate-recipient** is registered.

In the absence of this argument, the **recipient-assigned-alternate-recipient**, if any, remains unchanged.

8.4.1.1.1.7 *User-security-labels*

This argument contains the **security-labels** of the MTS-user, if they are to be changed. It may be generated by the MTS-user.

A **recipient-assigned-alternate-recipient** may be registered for any value of **user-security-labels**.

In the absence of this argument, the **user-security-labels** remain unchanged.

Note that some security-policies may only permit the **user-security-labels** to be changed in this way if a secure link is employed. Other local means of changing the **user-security-labels** in a secure manner may be provided.

8.4.1.1.1.8 *Default delivery control arguments*

The default control arguments are the same as the arguments of the delivery-control abstract-operation, and are defined in § 8.3.1.3.1. Except for **permissible-security-context**, they may be generated by the MTS-user.

The default controls are registered as arguments of the register abstract-operation. These defaults come into effect at the beginning of an association, and remain in effect until they are overridden by an invocation of the delivery-control abstract-operation.

The default control arguments shall not admit messages whose delivery are prohibited by the prevailing registered values of the **deliverable-encoded-information-types** argument, the **deliverable-content-types** argument or the **deliverable-maximum-content-length** argument.

8.4.1.1.2 *Results*

The register abstract-operation returns an empty result as indication of success.

8.4.1.1.3 *Abstract-errors*

Table 24/X.411 lists the abstract-errors that may disrupt the register abstract-operation, and for each abstract-error identifies the clause in which the abstract-error is defined.

TABLE 24/X.411

Register abstract-error

Abstract-error	Clause
Register-rejected	8.4.2.1

8.4.1.2 *Change-credentials*

The change-credentials abstract-operation enables the MTS-user to change the MTS-user's **credentials** held by the MTS, or enables the MTS to change the MTS's **credentials** held by the MTS-user.

The **credentials** are exchanged during the establishment of an association for the mutual authentication of identity of the MTS-user and the MTS.

The successful completion of the abstract-operation signifies that the **credentials** have been changed.

The disruption of the abstract-operation by an abstract-error indicates that the **credentials** have not been changed, either because the old **credentials** were incorrectly specified or that the new **credentials** are unacceptable.

8.4.1.2.1 *Arguments*

Table 25/X.411 lists the arguments of the change-credentials abstract-operation, and for each argument qualifies its presence and identifies the clause in which the argument is defined.

TABLE 25/X.411

Change-credentials arguments

Argument	Presence	Clause
Credential arguments		
Old-credentials	M	8.4.1.2.1.1
New-credentials	M	8.4.1.2.1.2

8.4.1.2.1.1 *Old-credentials*

This argument contains the current (old) **credentials** of the invoker of the abstract-operation, held by the performer of the abstract-operation. It shall be generated by the invoker of the abstract-operation.

If only simple-authentication is used, the **credentials** comprise a simple **password** associated with the **user-name**, or **MTA-name**, of the invoker.

If strong-authentication is used, the **credentials** comprise the **certificate** of the invoker, generated by a trusted source (e.g. a certification-authority), and supplied by the invoker.

8.4.1.2.1.2 *New-credentials*

This argument contains the proposed new **credentials** of the invoker of the abstract-operation, to be held by the performer of the abstract-operation. It shall be generated by the invoker of the abstract-operation.

The **new-credentials** shall be of the same type (i.e. simple or strong) as the **old-credentials**, as defined in § 8.4.1.2.1.1.

8.4.1.2.2 *Results*

The change-credentials abstract-operation returns an empty result as indication of success.

8.4.1.2.3 *Abstract-errors*

Table 26/X.411 lists the abstract-erros that may disrupt the change-credentials abstract-operation, and for each abstract-error identifies the paragraph in which the abstract-error is defined.

TABLE 26/X.411

Change-credentials abstract-errors

Abstract-error	Clause
New-credentials-unacceptable	8.4.2.2
Old-credential-incorrectly-specified	8.4.2.3

8.4.2 *Abstract-errors*

This section defines the following administration-port abstract-errors:

a) register-rejected
b) new-credentials-unacceptable
c) old-credentials-incorrectly-specified.

8.4.2.1 *Register-rejected*

The register-rejected abstract-error reports that the requested parameters cannot be registered because one or more are improperly specified.

The register-rejected abstract-error has no parameters.

8.4.2.2 *New-credentials-unacceptable*

The new-credentials-unacceptable abstract-error reports that the **credentials** cannot be changed because the **new-credentials** are unacceptable.

The new-credentials-unacceptable abstract-error has no parameters.

8.4.2.3 *Old-credentials-incorrectly-specified*

The old-credentials-incorrectly-specified abstract-error reports that the **credentials** cannot be changed because the current (**old-**) **credentials** were incorrectly specified.

The old-credentials-specified abstract-error has no parameters.

8.5 *Common parameter types*

This clause defines a number of common parameter types of the MTS abstract service.

8.5.1 *MTS-identifier*

MTS-identifiers are assigned by the MTS to distinguish between messages and probes at the MTS abstract service, and between messages, probes and reports within the MTS.

The **MTS-identifier** assigned to a message at a submission-port (**message-submission-identifier**) is identical to the corresponding **message-identifier** at a transfer-port and corresponding **message-delivery-identifier** at a delivery-port. Similarly, the **MTS-identifier** assigned to a probe at a submission-port (**probe-submission-identifier**) is identical to the corresponding **probe-identifier** at a transfer-port. **MTS-identifiers** are also assigned to reports at transfer-ports (**report-identifier**).

An **MTS-identifier** comprises:

- a **local-identifier** assigned by the MTA, which unambiguously identifies the related event within the MD;
- the **global-domain-identifier** of the MD, which ensures that the **MTS-identifier** is unambiguous throughout the MTS.

8.5.2 *Global-domain-identifier*

A **global-domain-identifier** unambiguously identifies an MD within the MHS.

A **global-domain-identifier** is used to ensure that an **MTS-identifier** is unambiguous throughout the MTS, and for identifying the source of a **trace-information-element.**

In the case of an ADMD, a **global-domain-identifier** consists of the **country-name** and the **administration-domain-name** of the MD. For a PRMD, it consists of the **contry-name** and the **administration-domain-name** of the associated ADMD, plus a **private-domain-identifier**. The **private-domain-identifier** is a unique identification of the PRMD, and may be identical to the PRMD's **private-domain-name**. As a national matter, this identification may be either relative to the country denoted by the **country-name** or relative to the associated ADMD.

Note 1 – The distinction between **private-domain-identifier** and **private-domain-name** has been retained for backward compatibility with Recommendation X.411 (1984). Often they will be identical.

Note 2 – In the **global-domain-identifier** of a PRMD, the **administration-domain-name** of the associated ADMD is optional ISO/IEC 10021-4.

8.5.3 *MTA-name*

An **MTA-name** is an identifier for an MTA that uniquely identifies the MTA within the MD to which it belongs.

8.5.4 *Time*

A **time** parameter is specified in terms of UTC (Coordinated Universal Time), and may optionally also contain an offset to UTC to convey the local time. The precision of the time of day is to either one second or one minute, determined by the generator of the parameter.

8.5.5 *OR-name*

An **OR-name** identifies the originator or recipient of a message according to the principles of naming and addressing described in Recommendation X.402.

At a submission-port, an **OR-name** comprises an **OR-address**, or a **directory-name**, or both (**OR-address-and-or-directory-name**). At all other types of port, an **OR-name** comprises an **OR-address** and, optionally, **directory-name (OR-address-and-optional-directory-name)**. A **directory-name** and an **OR-address** may each denote an individual originator or recipient, or a DL.

A **directory-name** is as defined in Recomendation X.501. The MTS uses the **directory-name** only when the **OR-address** is absent or invalid.

An **OR-address** comprises a number of **standard-attributes**, optionally a number of **extension-attributes**, and optionally a number of attributes defined by the MD to which the originator/recipient subscribes (**domain-defined-attributes**).

The **standard-** and **extension-attributes** used in an **OR-address** are selected from those defined in Recommendation X.402. Only those combinations of attributes explicitly defined in Recommendation X.402 can be used to form a valid **OR-address**.

8.5.6 *Encoded-information-types*

The **encoded-information-types** of a message are the kind(s) of information that appear in its **content**. Both basic **encoded-information-types** and externally-defined **encoded-information-types** may be specified, otherwise the **encoded-information-types** of a message are unspecified.

Externally-defined **encoded-information-types** are those to which object-identifiers are allocated by an appropriate authority. They include both standardised and private-defined **encoded-information-types**.

The basic **encoded-information-types** are those originally specified in the Recommendation X.411 (1984). The **undefined** type is any type other than the specified externally-defined **encoded-information-types** and other than the following types. The **telex** type is defined in Recommendation F.1. The **ia5-text** (teleprinter) type is defined in Recommendation T.50. The **g3-facsimile** type is defined in Recommendations T.4 and T.30. The **g4-class-1** type is defined in Recommendations T.5, T.6, T.400 and T.503. The **teletex** type is defined in Recommendations F.200, T.61 and T.60. The **videotex** type is defined in Recommendations T.100 and T.101. The **simple-formattable-document (sfd)** type is defined in Recommendation X.420 (1984) (Note that SFDs are no longer defined in any 1988 Recommendation). The **mixed-mode** type is defined in Recommendations T.400 and T.501.

Non-basic-parameters are defined for the **g3-facsimile, teletex, g4-class-1** and **mixed-mode** basic **encoded-information-types** for backwards compatibility with the Recommendation X.411 (1984) only. It is recommended that for each required combination of a basic **encoded-information-type** and a specific set of **non-basic-parameters**, an externally-defined **encoded-information-type** be defined and used in preference.

Note that **non-basic parameters** are likely to be removed from a future version of this Recommendation.

The **non-basic-parameters** for **g3-facsimile** correspond to the three- or four-octet Facsimile Information Field (FIF) conveyed by the Digital Command Signal (DCS) defined in Recommendation T.30. The parameters are: **two-dimensional, fine-resolution, unlimited-length, b4-length, a3-width, b4-width** and **uncompressed**.

The **non-basic-parameters** for **teletex** correspond to the non-basic terminal capability conveyed by the Command Document Start (CDS) defined in Recommendation T.62. The parameters are: optional **graphic-character-sets**, optional **control-character-sets**, optional **page-formats**, optional **miscellaneous-terminal-capabilities**, and a **private-use** parameter.

The **non-basic-parameters** for the **g4-class-1** and **mixed-mode** types specify optional resolution, optional graphic character sets, optional control character sets, and so on, which correspond to the parameters of the **presentation-capabilities** defined in Recommendations T.400, and T.503 and T.501.

Where **non-basic-parameters** are indicated, these parameters represent the logical 'OR' of the **non-basic-parameters** of each instance on the **encoded-information-type** in a message **content**. Thus, this parameter only serves to indicate whether there is **encoded-information-type** compatibility, or whether conversion is required. If conversion is required, the message **content** shall be inspected to determine which **non-basic-parameters** apply to any instance of the **encoded-information-type**.

8.5.7 *Certificate*

A **certificate** may be used to convey a verified copy of the public-asymmetric-encryption-key of the subject of the **certificate**.

A **certificate** contains the following parameters:

- **signature-algorithm-identifier**: an **algorithm-identifier** for the algorithm used by the certification-authority that issued the **certificate** to compute the **signature**;
- **issuer**: the **directory-name** of the certification-authority that issued the **certificate**;
- **validity**: a date and time of day before which the **certificate** should not be used, and a date and time of day after which the **certificate** should not be relied upon;
- **subject**: the **directory-name** of the subject of the **certificate**;
- **subject-public-keys**: one or more public-asymmetric-encryption-keys of the subject (each used in conjunction with an **algorithm** and a secret-asymmetric-encryption-key of the subject);
- **algorithms**: one or more **algorithm-identifiers**, each associated with a **subject-public-key**;
- **signature**: an asymmetrically encrypted, hashed version of the above parameters computed by the certification-authority that issued the **certificate** using the algorithm identified by the **signature-algorithm-identifier** and the certification-authority's secret-asymmetric-encryption-key.

If the originator and a recipient of a **certificate** are served by the same certification-authority, the recipient may use the certification-authority's public-asymmetric-encryption-key to validate the **certificate**, and derive the originator's public-asymmetric-encryption-key (**subject-public-key**).

If the originator and a recipient of a **certificate** are served by different certification-authorities, the recipient may require a return-certification-path to authenticate the originator's **certificate**. The **certificate** may therefore include an associated **certification-path**.

The **certification-path** may comprise a **forward-certification-path** which includes the certificate of the certification-authority that issued the **certificate**, together with the certificates of all of its superior certfication-authorities. The **forward-certification-path** may also include the certificates of other certification-authorities, cross-certified by either the certification-authority that issued the **certificate**, or any of its superior certification-authorities.

A recipient of the **certificate** may complete the required return-certification-path between the recipient and the originator of the **certificate** by appending the recipient's own reverse-certification-path to the **forward-certification-path** supplied by the originator, at a common-point-of-trust. The reverse-certification-path includes the reverse-certificate of the certification-authority of the recipient of the **certificate**, together with the reverse-certificate of all of it superior certification-authorities. The reverse-certification-path may also include the reverse-certificates of other certification-authorities, cross-certified by the certification-authority of the recipient of the **certificate**, or any of its superior certification authorities.

The return-certification-path thus formed allows the recipient of the **certificate** to validate each certificate in the return-certification-path in turn, to derive the public-asymmetric-encryption-key of the certification-authority that issued the **certificate**. The recipient may then use the public-asymmetric-encryption-key of the certification-authority that issued the **certificate** to validate the **certificate**, and derive the originator's public-asymmetric-encryption-key (**subject-public-key**).

The form of a **certificate**and a **certification-path** are further defined in Recommendation X.509.

Future versions of this Recommendation may define other key distribution techniques (e.g., based on symmetric-encryption-techniques).

8.5.8 *Token*

A **token** may be used to convey to the recipient of the **token** protected security-relevant information. The **token** provides authentication of public security-relevant information, and confidentiality and authentication of secret security-relevant information.

The type of a **token** is identified by a **token-type-identifier**. One type of **token** is currently defined by this Recommendation: an **asymmetric-token**. Other types of **token** may be defined by future versions of this Recommendation; for example, **tokens** based on symmetric-encryption techniques.

An **asymmetric-token** contains the following parameters:

- **signature-algorithm-identifier**: an **algorithm-identifier** for the algorithm used by the originator of the **token** to compute the **signature**;
- **recipient-name**: the **OR address and or directory name** of the intended-recipient of the **token**;
- **time**: the date and time of day when the **token** was generated;
- **signed-data**: public security-relevant information;
- **encryption-algorithm-identifier**: an **algorithm-identifier** for the algorithm used by the originator of the **token** to compute the **encrypted-data**;
- **encrypted-data**: secret security-relevant information encrypted by the originator of the **token** using the algorithm identified by the **encryption-algorithm-identifier** and the public-asymmetric-encryption-key of the intended-recipient of the **token**;
- **signature**: an asymmetrically encrypted, hashed version of the above parameters computed by the originator of the **token** using the algorithm identified by the **signature-algorithm-identifier** and the originator's secret-asymmetric-encryption-key.

The form of a **token** is further defined in Recommendation X.509.

8.5.9 *Security-label*

Security-labels may be used to associate security-relevant information with objects within the MTS.

Security-labels may be assigned to an object in line with the security-policy in force for that object. The security-policy may also define how **security-labels** are to be used to enforce that security-policy.

Within the scope of this Recommendation, **security-labels** may be associated with messages, probes and reports (see § 8.2.1.1.1.30), MTS-user (see § 8.4.1.1.1.7), MDs, MTAs and associations between an MTS-user and an MD(or MTA) (see § 8.1.1.1.1.4), or between MDs (or MTAs) (see § 12.1.1.1.1.4). Beyond the scope of this Recommendation, a security-policy may, as a local matter or by bilateral agreement, additionally assign **security-labels** to other objects within the MTS (e.g., secure routes).

A **security-label** comprises a set of **security-attributes**. The **security-attributes** may include a **security-policy-identifier**, a **security-classification**, a **privacy-mark**, and a set of **security-categories**.

A **security-policy-identifier** may be used to identify the security-policy in force to which the **security-label** relates.

If present, a **security-classification** may have one of a hierarchical list of values. The basic **security-classification** hierarchy is defined in this Recommendation, but the use of these values is defined by the security-policy in force. Additional values of **security-classification**, and their position in the hierarchy, may also be defined by a security-policy as a local matter or by bilateral agreement. The basic **security-classification** hierarchy is, in ascending order: **unmarked, unclassified, restricted, confidential, secret, top-secret**.

If present, a **privacy-mark** is a printable string. The content of the printable string may be defined by a security-policy, which may define a list of values to be used, or allow the value to be determined by the originator of the **security-label**. Examples of privacy-marks include **'IN CONFIDENCE'** and **'IN STRICTEST CONFIDENCE'**.

If present, the set of **security-categories** provide further restrictions within the context of a **security-classification** and/or **privacy-mark**, typically on a 'need-to-know' basis. The **security-categories** and their values may be defined by a security-policy as a local matter or by bilateral agreement. Examples of possible **security-categories** include caveats to the **security-classification** and/or **privacy-mark** (e.g., **'PERSONAL-'**, **'STAFF-'**, **'COMMERCIAL-'**, etc), closed-user-groups, codewords, etc.

8.5.10 *Algorithm-identifier*

An **algorithm-identifier** identifies an **algorithm** and any **algorithm-parameters** required by the **algorithm**.

An **algorithm-identifier** may be drawn from an international register of algorithms, or defined by bilateral agreement.

9 Message transfer system abstract syntax definition

The abstract-syntax of the MTS abstract service is defined in Figure 2/X.411.

The abstract-syntax of the MTS abstract service is defined using the abstract syntax notation (ASN.1) defined in Recommendation X.208, and the abstract service definition conventions defined in Recommendation X.407.

The abstract-syntax definition of the MTS abstract service has the following major parts:

- *Prologue:* declarations of the exports from, and imports to, the MTS abstract service module (Figure 2/X.411, Part 1).
- *Objects and ports:* definitions of the MTS and MTS-user objects, and their submission-, delivery- and administration-ports (Figure 2/X.411, Part 2).
- *MTS-bind and MTS-unbind:* definitions of the MTS-bind and MTS-unbind used to establish and release associations between an MTS-user and the MTS (Figure 2/X.411, Parts 3 to 4).
- *Submission port:* definitions of the submission-port abstract-operations: Message-submission, Probe-submission, Cancel-deferred-delivery and Submission-control; and their abstract-errors (Figure 2/X.411, Parts 5 to 7).
- *Delivery port:* definitions of the delivery-port abstract-operations: Message-delivery, Report-delivery and Delivery-control; and their abstract-errors (Figure 2/X.411, Parts 8 to 9).
- *Administration port:* definitions of the administration-port abstract-operations: Register and Change-credentials; and their abstract-errors (Figure 2/X.411, Parts 10 to 11).
- *Message submission envelope:* definition of the message-submission-envelope (Figure 2/X.411, Part 12).
- *Probe submission envelope:* definition of the probe-submission-envelope (Figure 2/X.411, Part 13).
- *Message delivery envelope:* definition of the message-delivery-envelope (Figure 2/X.411, Part 14).
- *Report delivery envelope:* definition of the report-delivery-envelope (Figure 2/X.411, Part 15).
- *Envelope fields:* definitions of envelope fields (Figure 2/X.411, Parts 16 to 19).
- *Extension fields:* definitions of extension-fields (Figure 2/X.411, Parts 20 to 28).
- *Common parameter types:* definitions of common parameter types (Figure 2/X.411, Parts 29 to 41).

Note 1 – The module implies a number of changes to the P3 protocol defined in Recommendation X.411 (1984). These changes are highlighted by means of underlining.

Note 2 – The module applies size constraints to variable-length data types using the SIZE subtyping extension of ASN.1. Violation of a size constraint constitutes a protocol violation.

9.1 *Criticality mechanism*

Each **extension-field** defined in Figure 2/X.411 (Parts 20 to 27) carries with it an indication of its **criticality** for submission, transfer and delivery. The criticality mechanism is designed to support controlled transparency of extended functions. A non-critical function may be ignored or discarded on delivery but shall not be discarded by a relaying MTA except when downgrading a message (see Recommendation X.419, Annex B), while a critical function must be known and performed correctly for normal procedure to continue.

In general, an argument of an abstract-operation marked critical for the port type shall be correctly handled by the performer of the abstract-operation, or an error reported in an appropriate way. The invoker of an abstract-operation shall also correctly handle any functions marked critical for the port type.

If the abstract-operation is one that reports an unsuccessful outcome, failure to correctly perform a critical function is reported by returning an unsupported-critical-function abstract-error. If an abstract-operation is not one that reports an unsuccessful outcome, an abstract-operation (e.g., a report) shall be invoked to convey the unsuccessful outcome of the previous operation (e.g., using the **unsupported-critical-function non-delivery diagnostic-code** of a report).

An extension that appears in the result of an abstract-operation shall not be marked critical for the port type.

In the case of **critical-for-submission**, the MTS shall correctly perform the procedures defined for a function marked as **critical-for-submission** in a message-submission or probe-submission abstract-operation, or shall return an unsupported-critical-function abstract-error.

In the case of **critical-for-transfer**, a receiving MTA shall correctly perform the procedures defined for a function in a message or probe marked as **critical-for-transfer**, or shall return a non-delivery-report with the **non-delivery-diagnostic-code** set to **unsupported-critical-function**. An MTA unable to support a function marked **critical-for-transfer** in a report shall discard the report (note that a local policy or agreement may require that this action be audited). An extension marked as **critical-for-transfer** that appears as an argument of a message-submission or probe-submission operation shall appear unchanged in a resulting message-transfer or probe-transfer operation at a transfer-port.

In the case of **critical-for-delivery**, a delivering-MTA shall correctly perform the procedures defined for a function marked **critical-for-delivery**, or shall not deliver the message or probe and shall return a non-delivery report with the **non-delivery-diagnostic-code** set to **unsupported-critical-function**. A recipient MTS-user shall correctly perform the procedures defined for a function marked as **critical-for-delivery** or shall return an unsupported-critical-function abstract-error. An extension marked as **critical-for-delivery** that appears as an argument of a message-submission or probe-submission operation shall appear unchanged in a resulting message-transfer or probe-transfer operation at a transfer port. An extension marked as **critical-for-delivery** that appears as an argument of a message-transfer or probe-transfer operation shall appear unchanged in any resulting message-transfer of probe-transfer operation at a transfer port.

An MTA generating a report shall not copy unsupported critical functions from the subject into the report. When generating a report, an MTA shall indicate the **criticality** (for transfer and/or delivery) of any supported functions copied from the subject into the report; the **criticality** of a function in a report may be different from its **criticality** in the subject.

If the MTA or MTS-user cannot correctly perform the procedures defined for a function marked "critical-for-delivery" in a report, then the report is discarded.

The procedures related to **extension-fields** and their **criticality** indications are further defined in § 14.

This Recommendation defines by means of the macro notation of ASN.1 the default setting of the **criticality** indication of **extension-fields** to be supplied by the originator of a message. The originator of a message or probe may choose, on a per-message basis, or in accordance with some local policy (e.g., a security-policy), to set the **criticality** indication of an extension-field to other than that defined in this Recommendation, either to relax or further constrain its **criticality**.

```
MTSAbstractService { joint-iso-ccitt mhs-motis(6) mts(3) modules(0) mts-abstract-service(1) }

DEFINITIONS IMPLICIT TAGS ::=

BEGIN

-- Prologue
-- Exports everything


IMPORTS
        -- Abstract service macros
        OBJECT, PORT, ABSTRACT-BIND, ABSTRACT-UNBIND, ABSTRACT-OPERATION, ABSTRACT-ERROR
                FROM AbstractServiceNotation { joint-iso-ccitt mhs-motis(6) asdc(2) modules(0)
                    notation(1) }


        -- MS Abstract service extension
        forwarding-request
                FROM MSAbstractService { joint-iso-ccitt mhs-motis(6) ms(4) modules(0)
                    abstract-service(1) }


        -- Object identifiers
        id-ot-mts, id-ot-mts-user,
        id-pt-submission, id-pt-delivery, id-pt-administration,
        id-att-physicalRendition-basic,
        id-tok-asymmetricToken
                FROM MTSObjectifiers { joint-iso-ccitt mhs-motis(6) mts(3) modules(0)
                    object-identifiers(0) }


        -- Directory definitions
        Name
                FROM InformationFramework { joint-iso-ccitt ds(5) modules(1)
                    information-framework(1) }
        PresentationAddress
                FROM SelectedAttributeTypes { joint-iso-ccitt ds(5) modules(1)
                    selectedAttributeTypes(5) }
        Certificates, AlgorithmIdentifier, ALGORITHM, SIGNED, SIGNATURE, ENCRYPTED
                FROM AuthenticationFramework { joint-iso-ccitt ds(5) modules(1)
                    authentication-framework(7) }
```

FIGURE 2/X.411 (Part 1 of 41)

Abstract syntax definition of the MTS abstract service

```
-- Upper bounds

ub-bit-options, ub-built-in-content-type, ub-built-in-encoded-information-types,
ub-common-name-length, ub-content-id-length, ub-content-length,
ub-content-types, ub-country-name-alpha-length, ub-country-name-numeric-length,
ub-dl-expansions, ub-domain-defined-attribute-value-length,
ub-domain-defined-attributes, ub-domain-defined-attribute-type-length,
ub-domain-name-length, ub-e163-4-number-length, ub-e163-4-subaddress-length,
ub-encoded-information-types, ub-extension-attributes, ub-extension-types,
ub-generation-qualifier-length, ub-given-name-length, ub-initials-length,
ub-integer-options, ub-labels-and-redirections, ub-local-id-length,
ub-mta-name-length, ub-mts-user-types, ub-numeric-user-id-length,
ub-organization-name-length, ub-organizational-unit-name-length,
ub-organizational-units, ub-password-length, ub-pds-name-length,
ub-pds-parameter-length, ub-pds-physical-address-lines, ub-postal-code-length, ub-privacy-mark-length,
ub-queue-size, ub-reason-codes, ub-recipients,
ub-recipient-number-for-advice-length, ub-redirections, ub-security-categories,
ub-security-labels, ub-security-problems, ub-supplementary-info-length,
ub-surname-length, ub-terminal-id-length, ub-tsap-id-length,
ub-informated-address-length, ub-x121-address-length

        FROM MTSUpperBounds { joint-iso-ccitt mhs-motis(6) mts(3) modules(0)
                upper-bounds(3) };
```

FIGURE 2/X.411 (Part 1 *bis* of 41)

Abstract syntax definition of the MTS abstract service

```
-- Objects

mTS OBJECT
        PORTS { submission [S], delivery [S], administration [S] }
        ::= id-ot-mts

mTSUser OBJECT
        PORTS { submission [C], delivery [C], administration [C] }
        ::= id-ot-mts-user


-- Ports

submission PORT
        CONSUMER INVOKES { MessageSubmission, ProbeSubmission, CancelDeferredDelivery }
        SUPPLIER INVOKES { SubmissionControl }
        ::= id-pt-submission

delivery PORT
        CONSUMER INVOKES { DeliveryControl }
        SUPPLIER INVOKES { MessageDelivery, ReportDelivery }
        ::= id-pt-delivery

administration PORT
        CONSUMER INVOKES { ChangeCredentials, Register }
        SUPPLIER INVOKES { ChangeCredentials }
        ::= id-pt-administration
```

FIGURE 2/X.411 (Part 2 of 41)

Abstract syntax definition of the MTS abstract service

```
-- MTS-bind and MTS-unbind

MTSBind ::= ABSTRACT-BIND
        TO { submission, delivery, administration }
        BIND
        ARGUMENT SET {
                initiator-name ObjectName,
                messages-waiting [1] EXPLICIT MessagesWaiting OPTIONAL,
                initiator-credentials [2] InitiatorCredentials,
                security-context [3] SecurityContext OPTIONAL }
        RESULT SET {
                responder-name ObjectName,
                messages-waiting [1] EXPLICIT MessagesWaiting OPTIONAL,
                responder-credentials [2] ResponderCredentials }
        BIND-ERROR INTEGER {
                busy (0)
                authentication-error (2),
                unacceptable-dialogue-mode (3),
                unacceptable-security-context (4) } (0. .ub-integer-options)

MTSUnbind ::=ABSTRACT-UNBIND
        FROM { submission, delivery, administration }


-- Association control parameters

ObjectName ::= CHOICE {
        mTS-userORAddressAndOptionalDirectoryName,
        mTA [0] MTAName,
        message-store [4] ORAddressAndOptionalDirectoryName }

MessagesWaiting ::= SET {
        urgent [0] DeliveryQueue,
        normal [1] DeliveryQueue,
        non-urgent [2] DeliveryQueue }

DeliveryQueue ::= SET {
        messages [0] INTEGER (0. .ub-queue-size),
        octets [1] INTEGER (0. .ub-content-length) OPTIONAL }

InitiatorCredentials ::= CHOICE {
        simple Password,
        strong [0] StrongCredentials (WITH COMPONENTS {
                ...,
                bind-token PRESENT }) }
```

FIGURE 2/X.411 (Part 3 of 41)

Abstract syntax definition of the MTS abstract service

```
ResponderCredentials ::= CHOICE {
        simple Password,
        strong [0] StrongCredentials (WITH COMPONENTS {
                bind-token }) }

Password ::= CHOICE {
        IA5String (SIZE (0. .ub-password-length))
        OCTET STRING (SIZE (0. .ub-password-length)) }

StrongCredentials ::= SET {
        bind-token [0] Token OPTIONAL,
        certificate [1] Certificates OPTIONAL }

Security Context ::= SET SIZE (1. .ub-security-labels) OF SecurityLabel
```

FIGURE 2/X.411 (Part 4 of 41)

Abstract syntax definition of the MTS abstract service

```
-- Submission port

MessageSubmission ::= ABSTRACT-OPERATION
        ARGUMENT SEQUENCE {
                envelope MessageSubmissionEnvelope,
                content Content }
        RESULT SET {
                message-submission-identifier MessageSubmissionIdentifier,
                message-submission-time [0] MessageSubmissionTime,
                content-identifier ContentIdentifier OPTIONAL,
                extensions [1] EXTENSIONS CHOSEN FROM {
                    originating-MTA-certificate,
                    proof-of-submission } DEFAULT {} }

        ERRORS {
                SubmissionControlViolated,
                ElementOfServiceNotSubscribed,
                OriginatorInvalid,
                RecipientImproperlySpecified,
                InconsistentRequest,
                SecurityError,
                UnsupportedCriticalFunction,
                RemoteBindError }

ProbeSubmission ::= ABSTRACT-OPERATION
        ARGUMENT
                envelope ProbeSubmissionEnvelope
        RESULT SET {
                probe-submission-identifier ProbeSubmissionIdentifier,
                probe-submission-time [0] ProbeSubmissionTime,
                content-identifier ContentIdentifier OPTIONAL }
        ERRORS {
                SubmissionControlViolated,
                ElementOfServiceNotSubscribed,
                OriginatorInvalid,
                RecipientImproperlySpecified,
                InconsistentRequest,
                SecurityError,
                UnsupportedCriticalFunction,
                RemoteBindError }

CancelDeferredDelivery :: ABSTRACT-OPERATION
        ARGUMENT
                message-submission-identifier MessageSubmissionIdentifier
        RESULT
        ERRORS {
                DeferredDeliveryCancellationRejected,
                MessageSubmissionIdentifierInvalid,
                RemoteBindError }
```

FIGURE 2/X.411 (Part 5 of 41)

Abstract syntax definition of the MTS abstract service

```
SubmissionControl ::= ABSTRACT-OPERATION
        ARGUMENT
                controls SubmissionControls
        RESULT
                waiting Waiting
        ERRORS {
                SecurityError,
                RemoteBindError }

SubmissionControlViolated ::= ABSTRACT-ERROR
        PARAMETER  NULL

ElementOfServiceNotSubscribed ::= ABSTRACT-ERROR
        PARAMETER  NULL

DeferredDeliveryCancellationRejected ::= ABSTRACT-ERROR
        PARAMETER  NULL

OriginatorInvalid ::= ABSTRACT-ERROR
        PARAMETER  NULL

RecipientImproperlySpecified ::= ABSTRACT-ERROR
        PARAMETER
                improperly-specified-recipients SEQUENCE SIZE (1. .ub-recipients OF
                    ORAddressAndOptionalDirectoryName

MessageSubmissionIdentifierInvalid ::= ABSTRACT-ERROR
        PARAMETER  NULL

InconsistentRequest ::= ABSTRACT-ERROR
        PARAMETER  NULL

SecurityError ::= ABSTRACT-ERROR
        PARAMETER
                security-problem SecurityProblem

SecurityProblem ::= INTEGER (0. .ub-security-problems)

UnsupportedCriticalFunction ::= ABSTRACT-ERROR
        PARAMETER  NULL

RemoteBindError ::= ABSTRACT-ERROR
        PARAMETER  NULL
```

FIGURE 2/X.411 (Part 6 of 41)

Abstract syntax definition of the MTS abstract service

```
-- Submision port parameters

MessageSubmissionIdentifier ::= MTSIdentifier

MessageSubmissionTime ::= Time

ProbeSubmissionIdentifier ::= MTSIdentifier

ProbeSubmissionTime ::= Time

SubmissionControls ::= Controls (WITH COMPONENTS {
        permissible-content-types ABSENT
        permissible-encoded-information-types ABSENT })

Waiting ::= SET {
        waiting-operations [0] Operations DEFAULT { },
        waiting-messages [1] WaitingMessages DEFAULT { },
        waiting-content-types [2] SET SIZE (0. .ub-content-types) OF ContentType DEFAULT { },
        waiting-encoded-information-types EncodedInformationTypes OPTIONAL }

Operations ::= BIT STRING {
        probe-submission-or-report-delivery (0),
        message-submission-or-message-delivery (1) } (SIZE (0. .ub-bit-options))
        --holding 'one', not-holding 'zero'.

WaitingMessages ::= BIT STRING {
        long-content (0),
        low-priority (1),
        other-security-labels (2) } (SIZE (0. .ub-bit-options))
```

FIGURE 2/X.411 (Part 7 of 41)

Abstract syntax definition of the MTS abstract service

```
-- Delivery port

MessageDelivery ::= ABSTRACT-OPERATION
        ARGUMENT SEQUENCE {
                COMPONENTS OF MessageDeliveryEnvelope,
                content Content }
        RESULT SET {
                recipient-certificate [0] RecipientCertificate OPTIONAL,
                proof-of-delivery [1] ProofOfDelivery OPTIONAL } DEFAULT { }
        ERRORS {
                DeliveryControlViolated,
                SecurityError,
                UnsupportedCriticalFunction }

ReportDelivery ::= ABSTRACT-OPERATION
        ARGUMENT SET {
                COMPONENTS OF ReportDeliveryEnvelope,
                returned-content [0] Content OPTIONAL }
        RESULT
        ERRORS {
                DeliveryControlViolated,
                SecurityError,
                UnsupportedCriticalFunction }

DeliveryControl ::= ABSTRACT-OPERATION
        ARGUMENT
                controls DeliveryControls
        RESULT
                waiting Waiting
        ERRORS {
                ControlViolatesRegistration,
                SecurityError }

DeliveryControlViolated ::= ABSTRACT-ERROR
        PARAMETER  NULL

ControlViolatesRegistration ::= ABSTRACT-ERROR
        PARAMETER  NULL


-- SecurityError – defined in Figure 2/X.411, Part 6 of 41

-- UnsupportedCriticalFunction – defined in Figure 2/X.411, Part 6 of 41
```

FIGURE 2/X.411 (Part 8 of 41)

Abstract syntax definition of the MTS abstract service

```
-- Delivery port parameters

RecipientCertificate ::= Certificates

ProofOfDelivery ::= SIGNATURE SEQUENCE{
        algorithm-identifier ProofOfDeliveryAlgorithmIdentifier,
        delivery-timeMessageDeliveryTime,
        this-recipient-name ThisRecipientName,
        originally-intended-recipient-name OriginallyIntendedRecipientName OPTIONAL,
        content Content,
        content-identifier ContentIdentifier OPTIONAL,
        message-security-label MessageSecurityLabel OPTIONAL}

ProofOfDeliveryAlgorithmIdentifier ::= AlgorithmIdentifier

DeliveryControls ::= Controls

Controls ::= SET{
        restrict [0] BOOLEAN DEFAULT TRUE,
        -- update 'TRUE', remove 'FALSE'
        permissible-operations [1] Operations OPTIONAL,
        permissible-maximum-content-length [2] ContentLength OPTIONAL,
        permissible-lowest-priority Priority OPTIONAL,
        permissible-content-types [4] SET SIZE (1. .ub-content-types) OF ContentType OPTIONAL,
        permissible-encoded-information-types EncodedInformationTypes OPTIONAL,
        permissible-security-context [5] SecurityContext OPTIONAL}

-- Note – The tags [0], [1] and [2] are altered for the register operation only.
```

FIGURE 2/X.411 (Part 9 of 41)

Abstract syntax definition of the MTS abstract service

```
-- Administration port

Register ::= ABSTRACT-OPERATION
        ARGUMENT SET{
                user-name UserName OPTIONAL,
                user-address [0] UserAddress OPTIONAL,
                deliverable-encoded-information-types EncodedInformationTypes OPTIONAL,
                deliverable-maximum-content-length [1] EXPLICIT ContentLength OPTIONAL,
                default-delivery-controls [2] EXPLICIT DefaultDeliveryControls OPTIONAL,
                deliverable-content-types [3] SET SIZE (1 . .ub-content-types) OF ContentType OPTIONAL,
                labels-and-redirections [4] SET SIZE (1. .ub-labels-and-redirections) OF
                    LabelAndRedirection OPTIONAL }
        RESULT
        ERRORS{
                RegisterRejected }

ChangeCredentials ::= ABSTRACT-OPERATION
        ARGUMENT SET{
                old-credentials [0] Credentials,
                new-credentials [1] Credentials -- same CHOICE as for old-credentials -- }
        RESULT
        ERRORS{
                NewCredentialsUnacceptable,
                OldCredentialsIncorrectlySpecified }

RegisterRejected ::= ABSTRACT-ERROR
        PARAMETER  NULL

NewCredentialsUnacceptable ::= ABSTRACT-ERROR
        PARAMETER  NULL

OldCredentialsIncorrectlySpecified ::= ABSTRACT-ERROR
        PARAMETER  NULL
```

FIGURE 2/X.411 (Part 10 of 41)

Abstract syntax definition of the MTS abstract service

```
-- Administration port parameters

UserName ::= ORAddressAndOptionalDirectoryName

UserAddress ::= CHOICE {
        x121 [0] SEQUENCE {
                x121-address NumericString (Size (1. .ub-x121-address-length)) OPTIONAL,
                tsap-id PrintableString (SIZE (1. .ub-tsap-id-length)) OPTIONAL },
        presentation [1] PSAPAddress }

PSAPAddress ::= PresentationAddress

DefaultDeliveryControls ::= Controls (WITH COMPONENTS {
        ...,
        permissible-security-context ABSENT })

Credentials ::= CHOICE {
        simple Password,
        strong [0] StrongCredentials (WITH COMPONENTS {
                certificate }) }

LabelAndRedirection ::= SET {
        user-security-label [0] UserSecurityLabel OPTIONAL,
        recipient-assigned-alternate-recipient [1] RecipientAssignedAlternateRecipient OPTIONAL }

UserSecurityLabel ::= SecurityLabel

RecipientAssignedAlternateRecipient ::= ORAddressAndOptionalDirectoryName
```

FIGURE 2/X.411 (Part 11 of 41)

Abstract syntax definition of the MTS abstract service

```
-- Message submission envelope

MessageSubmissionEnvelope ::= SET{
        COMPONENTS OF PerMessageSubmissionFields,
        per-recipient-fields [1] SEQUENCE SIZE (1. .ub-recipients) OF
                PerRecipientMessageSubmissionFields }

PerMessageSubmissionFields ::= SET{
        originator-name OriginatorName,
        original-encoded-information-types OriginalEncodedInformationTypes OPTIONAL,
        content-type ContentType,
        content-identifier ContentIdentifier OPTIONAL,
        priority Priority DEFAULT normal,
        per-message-indicators PerMessageIndicators DEFAULT {},
        deferred-delivery-time [0] DeferredDeliveryTime OPTIONAL,
        extensions [2] PerMessageSubmissionExtensions DEFAULT {}}

PerMessageSubmissionExtensions ::= EXTENSIONS CHOSEN FROM{
        recipient-reassignment-prohibited,
        dl-expansion-prohibited,
        conversion-with-loss-prohibited,
        latest-delivery-time,
        originator-return-address,
        originator-certificate,
        content-confidentiality-algorithm-identifier,
        message-origin-authentication-check,
        message-security-label,
        proof-of-submission-request,
        content-correlator,
        forwarding-request -- for MS Abstract Service only -- }

PerRecipientMessageSubmissionFields ::= SET{
        recipient-name RecipientName,
        originator-report-request [0] OriginatorReportRequest,
        explicit-conversion [1] ExplicitConversion OPTIONAL,
        extensions [2] PerRecipientMessageSubmissionExtensions DEFAULT {}}

PerRecipientMessageSubmissionExtensions ::= EXTENSIONS CHOSEN FROM{
        originator-requested-alternate-recipient,
        requested-delivery-method,
        physical-forwarding-prohibited,
        physical-forwarding-address-request,
        physical-delivery-modes,
        registered-mail-type,
        recipient-number-for-advice,
        physical-rendition-attributes,
        physical-delivery-report-request,
        message-token,
        content-integrity-check,
        proof-of-delivery-request }
```

FIGURE 2/X.411 (Part 12 of 41)

Abstract syntax definition of the MTS abstract service

```
-- Probe submission envelope

ProbeSubmissionEnvelope ::= SET{
        COMPONENTS OF PerProbeSubmissionFields,
        per-recipient-fields [3] SEQUENCE SIZE (1. .ub-recipients) OF
                PerRecipientProbeSubmissionFields}

PerProbeSubmissionFields ::= SET{
        originator-name OriginatorName,
        original-encoded-information-types OriginalEncodedInformationTypes OPTIONAL
        content-type ContentType,
        content-identifier ContentIdentifier OPTIONAL,
        content-length [0] ContentLength OPTIONAL,
        per-message-indicators PerMessageIndicators DEFAULT {},
        extensions [2] EXTENSIONS CHOSEN FROM{
                recipient-reassignment-prohibited,
                dl-expansion-prohibited,
                conversion-with-loss-prohibited,
                originator-certificate,
                message-security-label,
                content-correlator,
                probe-origin-authentication-check} DEFAULT {}}

PerRecipientProbeSubmissionFields ::= SET{
        recipient-name RecipientName,
        originator-report-request [0] OriginatorReportRequest,
        explicit-conversion [1] ExplicitConversion OPTIONAL,
        extensions [2] EXTENSIONS CHOSEN FROM{
                originator-requested-alternate-recipient,
                requested-delivery-method
                physical-rendition-attributes} DEFAULT {}}
```

FIGURE 2/X.411 (Part 13 of 41)

Abstract syntax definition of the MTS abstract service

```
-- Message delivery envelope

MessageDeliveryEnvelope ::= SEQUENCE{
        message-delivery-identifier MessageDeliveryIdentifier,
        message-delivery-time MessageDeliveryTime,
        other-fields OtherMessageDeliveryFields }

OtherMessageDeliveryFields ::= SET{
        content-type DeliveredContentType,
        originator-name OriginatorName,
        original-encoded-information-types [1] OriginalEncodedInformationTypes OPTIONAL,
        priority Priority DEFAULT normal,
        delivery-flags [2] DeliveryFlags OPTIONAL,
        other-recipient-names [3] OtherRecipientNames OPTIONAL,
        this-recipient-name [4] ThisRecipientName,
        originally-intended-recipient-name [5] OriginallyIntendedRecipientName OPTIONAL,
        converted-encoded-information-types [6] ConvertedEncodedInformationTypes OPTIONAL,
        message-submission-time [7] MessageSubmissionTime,
        content-identifier [8] ContentIdentifier OPTIONAL,
        extensions [9] EXTENSIONS CHOSEN FROM{
                conversion-with-loss-prohibited,
                requested-delivery-method,
                physical-forwarding-prohibited,
                physical-forwarding-address-request,
                physical-delivery-modes,
                registered-mail-type,
                recipient-number-for-advice,
                physical-rendition-attributes,
                originator-return-address,
                physical-delivery-report-request,
                originator-certificate,
                message-token,
                content-confidentiality-algorithm-identifier,
                content-integrity-check,
                message-origin-authentication-check,
                message-security-label,
                proof-of-delivery-request,
                redirection-history,
                dl-expansion-history } DEFAULT {}}
```

FIGURE 2/X.411 (Part 14 of 41)

Abstract syntax definition of the MTS abstract service

```
-- Report delivery envelope

ReportDeliveryEnvelope ::= SET{
        COMPONENTS OF PerReportDeliveryFields,.
        per-recipient-fields SEQUENCE SIZE (1..ub-recipients) OF PerRecipientReportDeliveryFields }

PerReportDeliveryFields ::= SET{
        subject-submission-identifier SubjectSubmissionIdentifier,
        content-identifier ContentIdentifer OPTIONAL,
        content-type ContentType OPTIONAL,
        original-encoded-information-types OriginalEncodedInformationTypes OPTIONAL,
        extension [1] EXTENSIONS CHOSEN FROM{
                message-security-label,
                content-correlator,
                originator-and-DL-expansion-history,
                reporting-DL-name,
                reporting-MTA-certificate,
                report-origin-authentication-check} DEFAULT {}}

PerRecipientReportDeliveryFields ::= SET{
        actual-recipient-name [0] ActualRecipientName,
        report-type [1] ReportType,
        converted-encoded-information-types ConvertedEncodedInformationTypes OPTIONAL,
        originally-intended-recipient-name [2] OriginallyIntendedRecipientName OPTIONAL,
        supplementary-information [3] SupplementaryInformation OPTIONAL,
        extensions [4] EXTENSIONS CHOSEN FROM {
                redirection-history,
                physical-forwarding-address,
                recipient-certificate,
                proof-of-delivery} DEFAULT {}}

ReportType ::= CHOICE{
        delivery [0] DeliveryReport,
        non-delivery [1] NonDeliveryReport }

DeliveryReport ::= SET{
        message-delivery-time [0] MessageDeliveryTime,
        type-of-MTS-user [1] TypeOfMTSUser DEFAULT public }

NonDeliveryReport ::= SET{
        non-delivery-reason-code [0] NonDeliveryReasonCode,
        non-delivery-diagnostic-code [1] NonDeliveryDiagnosticCode OPTIONAL }
```

FIGURE 2/X.411 (Part 15 of 41)

Abstract syntax definition of the MTS abstract service

```
-- Envelope fields

OriginatorName :: = ORAddressAndOrDirectoryName

OriginalEncodedInformationTypes ::= EncodedInformationTypes

ContentType ::= CHOICE {
        built-in BuiltInContentType,
        external ExternalContentType }

BuiltInContentType ::= [APPLICATION 6] INTEGER {
        unidentified (0),
        external (1),                          -- identified by the object-identifier of the EXTERNAL content
        interpersonal-messaging-1984 (2),
        interpersonal-messaging-1988 (22) } (0. .ub-built-in-content-type)

ExternalContentType ::= OBJECT IDENTIFIER

DeliverContentType ::= CHOICE {
        built-in [0] BuiltInContentType,
        external ExternalContentType }

ContentIdentifier ::= [APPLICATION 10] PrintableString (SIZE (1. .ub-content-id-length))

PerMessageIndicators ::= [APPLICATION 8] BIT STRING {
        disclosure-of-recipients (0),          -- disclosure-of-recipients-allowed 'one',
                                               -- disclosure-of-recipient-prohibited 'zero';
                                               -- ignored for Probe-submission
        implicit-conversion-prohibited (1),    -- implicit-conversion-prohibited 'one';
                                               -- implicit-conversion-allowed 'zero'
        alternate-recipient-allowed (2),       -- alternate-recipient-allowed 'one',
                                               -- alternate-recipient-prohibited 'zero'
        content-return-request (3)             -- content-return-requested 'one',
                                               -- content-return-not-requested 'zero';
                                               -- ignored for Probe-submission -- }
        (SIZE (0. .ub-bit-options))

RecipientName ::= ORAddressAndOrDirectoryName

OriginatorReportRequest ::= BIT STRING {
        report (3),
        non-delivery-report (4)
        -- at most one bit shall be 'one':
        -- report bit 'one' requests a 'report';
        -- non-delivery-report bit 'one' requests a 'non-delivery-report';
        -- both bits 'zero' requests 'no-report' -- } (SIZE (0. .ub-bit-options))
```

FIGURE 2/X.411 (Part 16 of 41)

Abstract syntax definition of the MTS abstract service

```
ExplicitConversion ::= INTEGER {
        ia5-text-to-teletex (0),
        teletex-to-telex (1),
        telex-to-ia5-text (2),
        telex-to-teletex (3),
        telex-to-g4-class-1 (4),
        telex-to-videotex (5),
        ia5-text-to-telex (6),
        telex-to-g3-facsimile (7),
        ia5-text-to-g3-facsimile (8),
        ia5-text-to-g4-class-1 (9),
        ia5-text-to-videotex (10),
        teletex-to-ia5-text (11),
        teletex-to-g3-facsimile (12),
        teletex-to-g4-class-1 (13),
        teletex-to-videotex (14),
        videotex-to-telex (15),
        videotex-to-ia5-text (16),
        videotex-to-teletex (17) } (0. .ub-integer-options)

DeferredDeliveryTime ::= Time

Priority ::= [APPLICATION 7] ENUMERATED {
        normal (0),
        non-urgent (1),
        urgent (2) }

ContentLength ::= INTEGER (0. .ub-content-length)

MessageDeliveryIdentifier ::= MTSIdentifier

MessageDeliveryTime ::= Time

DeliveryFlags ::= BIT STRING {
        implicit-conversion-prohibited (1)          -- implicit-conversion-prohibited 'one',
                                                    -- implicit-conversion-allowed 'zero' -- }
        (SIZE (0. .ub-bit-options))

OtherRecipientNames ::= SEQUENCE SIZE (1. .ub-recipients) OF OtherRecipientName

OtherRecipientName ::= OrAddressAndOrDirectoryName

ThisRecipientName ::= OrAddressAndOrDirectoryName

OriginallyIntendedRecipientName ::= OrAddressAndOrDirectoryName
```

FIGURE 2/X.411 (Part 17 of 41)

Abstract syntax definition of the MTS abstract service

```
ConvertedEncodedInformationTypes ::= EncodedInformationTypes

SubjectSubmissionIdentifier ::= MTSIdentifier

ActualRecipientName ::= ORAddressAndOrDirectoryName

TypeOfMTSUser ::= INTEGER {
        public (0),
        private (1),
        ms (2),
        dl (3),
        pdau (4),
        physical-recipient (5),
        other (6) } (0. .ub-mts-user-types)

NonDeliveryReasonCode ::= INTEGER {
        transfer-failure (0),
        unable-to-transfer (1),
        conversion-not-performed (2),
        physical-rendition-not-performed (3),
        physical-delivery-not-performed (4),
        restricted-delivery (5),
        directory-operation-unsuccessful (6) } (0. .ub-reason-codes)

NonDeliveryDiagnosticCode ::= INTEGER {
        unrecognised-OR-name (0),
        ambiguous-OR-name (1),
        mts-congestion (2),
        loop-detected (3),
        recipient-unavailable (4),
        maximum-time-expired (5),
        encoded-information-types-unsupported (6),
        content-too-long (7),
        conversion-impratical (8),
        implicit-conversion-prohibited (9),
        implicit-conversion-not-subscribed (10),
        invalid-arguments (11),
        content-syntax-error (12),
        size-constraint-violation (13),
        protocol-violation (14),
        content-type-not-supported (15),
        too-many-recipients (16),
        no-bilateral-agreement (17),
        unsupported-critical-function (18),

        -- continued
```

FIGURE 2/X.411 (Part 18 of 41)

Abstract syntax definition of the MTS abstract service

-- continued

```
conversion-with-loss-prohibited (19),
line-too-long (20),
page-split (21),
pictorial-symbol-loss (22),
punctuation-symbol-loss (23),
alphabetic-character-loss (24),
multiple-information-loss (25),
recipient-reassignment-prohibited (26),
redirection-loop-detected (27),
dL-expansion-prohibited (28),
no-DL-submit-permission (29),
dl-expansion-failure (30),
physical-rendition-attributes-not-supported (31),
undeliverable-mail-physical-delivery-address-incorrect (32),
undeliverable-mail-physical-delivery-office-incorrect-or-invalid (33),
undeliverable-mail-physical-delivery-address-incomplete (34),
undeliverable-mail-recipient-unknown (35),
undeliverable-mail-recipient-deceased (36),
undeliverable-mail-organization-expired (37),
undeliverable-mail-recipient-refused-to-accept (38),
undeliverable-mail-recipient-did-not-claim (39),
undeliverable-mail-recipient-changed-address-permanently (40),
undeliverable-mail-recipient-changed-address-temporarily (41),
undeliverable-mail-recipient-changed-temporary-address (42),
undeliverable-mail-new-address-unknown (43),
undeliverable-mail-recipient-did-not-want-forwarding (44),
undeliverable-mail-originator-prohibited-forwarding (45),
secure-messaging-error (46),
unable-to-downgrade (47) } (0. .ub-diagnostic-codes)
SupplementaryInformation ::= PrintableString (SIZE (1. .ub-supplementary-info-length))
```

FIGURE 2/X.411 (Part 19 of 41)

Abstract syntax definition of the MTS abstract service

```
-- Extension fields

ExtensionField ::= SEQUENCE {
        type [0] EXTENSION,
        criticality [1] Criticality DEFAULT {},
        value [2] AND DEFINED BY type DEFAULT NULL NULL }

Criticality ::= BIT STRING {
        for-submission (0),
        for-transfer (1),
        for-delivery (2) } (SIZE (0. .ub-bit-options))        -- critical 'one', non-critical 'zero'

EXTENSIONS MACRO ::=
BEGIN

TYPE NOTATION ::= "CHOSEN FROM" "{" ExtensionList "}"
VALUE NOTATION ::= Value (VALUE SET OF ExtensionField      -- each of a different type --)

ExtensionList ::= Extension "," ExtensionList | Extension | empty
Extension ::= value (EXTENSION)

END  -- of EXTENSIONS

EXTENSION MACRO ::=
BEGIN

TYPE NOTATION ::= DataType Critical | empty
VALUE NOTATION ::= value (VALUE ExtensionType)

DataType ::= type (X) Default | empty
Default ::= "DEFAULT" value (X) | empty
Critical ::= "CRITICAL FOR" CriticalityList | empty
CriticalityList ::= Criticality | CriticalityList "," Criticality
Criticality ::= "SUBMISSION" | "TRANSFER" | "DELIVERY"

END  -- of EXTENSION

ExtensionType ::= INTEGER (0. .ub-extension-types)

recipient-reassignment-prohibited EXTENSION
        RecipientReassignmentProhibited DEFAULT recipient-reassignment-allowed
        CRITICAL FOR DELIVERY
        ::= 1
```

FIGURE 2/X.411 (Part 20 of 41)

Abstract syntax definition of the MTS abstract service

```
RecipientReassignmentProhibited ::= ENUMERATED {
        recipient-reassignment-allowed (0),
        recipient-reassignment-prohibited (1) }

originator-requested-alternate-recipient EXTENSION
        OriginatorRequestedAlternateRecipient
        CRITICAL FOR SUBMISSION
        ::= 2

OriginatorRequestedAlternateRecipient ::= ORAddressAndOrDirectoryName

-- OriginatorRequestedAlternateRecipient as defined here differs from the
-- field of the same name in Figure 4/X.411, since, on submission the
-- OR-address need not be present, but on transfer the OR-address
-- must be present.

dl-expansion-prohibited EXTENSION
        DLExpansionProhibited DEFAULT dl-expansion-allowed
        CRITICAL FOR DELIVERY
        ::= 3

DLExpansionProhibited ::= ENUMERATED {
        dl-expansion-allowed (0),
        dl-expansion-prohibited (1) }

conversion-with-loss-prohibited EXTENSION
        ConversionWithLossProhibited DEFAULT conversion-with-loss-allowed
        CRITICAL FOR DELIVERY
        ::= 4

ConversionWithLossProhibited ::= ENUMERATED {
        conversion-with-loss-allowed (0),
        conversion-with-loss-prohibited (1) }

latest-delivery-time EXTENSION
        LatestDeliveryTime
        CRITICAL FOR DELIVERY
        ::= 5

LatestDeliveryTime ::= Time

requested-delivery-method EXTENSION
        RequestedDeliveryMethod DEFAULT any-delivery-method
        CRITICAL FOR DELIVERY
        ::= 6
```

FIGURE 2/X.411 (Part 21 of 41)

Abstract syntax definition of the MTS abstract service

```
RequestedDeliveryMethod ::= SEQUENCE OF INTEGER { -- each different in order of preference,
                                                   most preferred first

        any-delivery-method (0),
        mhs-delivery (1),
        physical-delivery (2),
        telex-delivery (3),
        teletex-delivery (4),
        g3-facsimile-delivery (5),
        g4-facsimile-delivery (6),
        ia5-terminal-delivery (7),
        videotex-delivery (8),
        telephone-delivery (9) }  (0. .ub-integer-options)

physical-forwarding-prohibited EXTENSION
        PhysicalForwardingProhibited DEFAULT physical-forwarding-allowed
        CRITICAL FOR DELIVERY
        ::= 7

PhysicalForwardingProhibited ::= ENUMERATED {
        physical-forwarding-allowed (0),
        physical-forwarding-prohibited (1) }

physical-forwarding-address-request EXTENSION
        PhysicalForwardingAddressRequest DEFAULT physical-forwarding-address-not-requested
        CRITICAL FOR DELIVERY
        ::= 8

PhysicalForwardingAddressRequest ::= ENUMERATED {
        physical-forwarding-address-not-requested (0),
        physical-forwarding-address-requested (1) }

physical-delivery-modes EXTENSION
        PhysicalDeliveryModes DEFAULT ordinary-mail
        CRITICAL FOR DELIVERY
        ::= 9

PhysicalDeliveryModes ::= BIT STRING {
        ordinary-mail (0),
        special-delivery (1),
        express-mail (2),
        counter-collection (3),
        counter-collection-with-telephone-advice (4),
        counter-collection-with-telex-advice (5),
        counter-collection-with-teletex-advice (6),
        bureau-fax-delivery (7)
        -- bits 0 to 6 are mutually exclusive
        -- bit 7 can be set with any of bits 0 to 6 -- } (SIZE (0. .ub-bit-options))
```

FIGURE 2/X.411 (Part 22 of 41)

Abstract syntax definition of the MTS abstract service

```
registered-mail-type EXTENSION
        RegisteredMailType DEFAULT non-registered-mail
        CRITICAL FOR DELIVERY
        ::= 10

RegisteredMailType ::= INTEGER
        non-registered-mail (0),
        registered-mail (1),
        registered-mail-to-addresse-in-person (2) } (0. .ub-integer-options)

recipient-number-for-advice EXTENSION
        RecipientNumberForAdvice
        CRITICAL FOR DELIVERY
        ::= 11

RecipientNumberForAdvice ::= TeletexString (SIZE (1. .ub-recipient-number-for-advice-length)

physical-rendition-attributes EXTENSION
        PhysicalRenditionAttributes DEFAULT id-att-physicalRendition-basic
        CRITICAL FOR DELIVERY
        :: = 12

PhysicalRenditionAttributes ::= OBJECT IDENTIFIER

originator-return-address EXTENSION
        OriginatorReturnAddress
        CRITICAL FOR DELIVERY
        ::= 12

OriginatorReturnAddress ::= ORAddress

physical-delivery-report-request EXTENSION
        PhysicalDeliveryReportRequest DEFAULT return-of-undeliverable-mail-by-PDS
        CRITICAL FOR DELIVERY
        ::= 14

PhysicalDeliveryReportRequest::= INTEGER {
        return-of-undeliverable-mail-by-PDS (0),
        return-of-notification-by-PDS (1),
        return-of-notification-by-MHS (2),
        return-of-notification-by-MHS-and-PDS (3) } (0. .ub-integer-options)
```

FIGURE 2/X.411 (Part 23 of 41)

Abstract syntax definition of the MTS abstract service

```
originator-certificate EXTENSION
        OriginatorCertificate
        CRITICAL FOR DELIVERY
        ::= 15

OriginatorCertificate ::= Certificates

message-token EXTENSION
        MessageToken
        ::= 16

MessageToken ::= Token

content-confidentiality-algorithm-identifier EXTENSION
        ContentConfidentialityAlgorithmIdentifier
        ::= 17

ContentConfidentialityAlgorithmIdentifier ::= AlgorithmIdentifier

content-integrity-check EXTENSION
        ContentIntegrityCheck
        ::= 18

ContentIntegrityCheck ::= SIGNATURE SEQUENCE {
        algorithm-identifier ContentIntegrityAlgorithmIdentifier,
        content Content }

ContentIntegrityAlgorithmIdentifier ::= AlgorithmIdentifier

message-origin-authentication-check EXTENSION
        MessageOriginAuthenticationCheck
        CRITICAL FOR DELIVERY
        ::= 19

MessageOriginAuthenticationCheck ::= SIGNATURE SEQUENCE {
        algorithm-identifier MessageOriginAuthenticationAlgorithmIdentifier,
        content Content
        content-identifier ContentIdentifier OPTIONAL,
        message-security-label MessageSecurityLabel OPTIONAL }

MessageOriginAuthenticationAlgorithmIdentifier ::= AlgorithmIdentifier
```

FIGURE 2/X.411 (Part 24 of 41)

Abstract syntax definition of the MTS abstract service

```
message-security-label EXTENSION
        MessageSecurityLabel
        CRITICAL FOR DELIVERY
        ::= 20

MessageSecurityLabel ::= SecurityLabel

proof-of-submission-request EXTENSION
        ProofOfSubmissionRequest DEFAULT proof-of-submission-not-requested
        CRITICAL FOR SUBMISSION
        ::= 21

ProofOfSubmissionRequest ::= ENUMERATED {
        proof-of-submission-not-requested (0),
        proof-of-submission-requested (1) }

proof-of-delivery-request EXTENSION
        ProofOfDeliveryRequest DEFAULT proof-of-delivery-not-requested
        CRITICAL FOR DELIVERY
        ::= 22

ProofOfDeliveryRequest ::= ENUMERATED {
        proof-of-delivery-not-requested (0),
        proof-of-delivery-requested (1) }

content-correlator EXTENSION
        ContentCorrelator
        ::= 23

ContentCorrelator ::= ANY       -- maximum ub-content-correlator-length octets including all encoding

probe-origin-authentication-check EXTENSION
        ProbeOriginAuthenticationCheck
        CRITICAL FOR DELIVERY
        ::= 24

ProbeOriginAuthenticationCheck ::= SIGNATURE SEQUENCE {
        algorithm-identifier ProbeOriginAuthenticationAlgorithmIdentifier,
        content-identifier ContentIdentifier OPTIONAL,
        message-security-label MessageSecurityLabel OPTIONAL }

ProbeOriginAuthenticationAlgorithmIdentifier ::= AlgorithmIdentifier
```

FIGURE 2/X.411 (Part 25 of 41)

Abstract syntax definition of the MTS abstract service

```
redirection-history EXTENSION
        RedirectionHistory
        ::= 25

RedirectionHistory ::= SEQUENCE SIZE (1. .ub-redirections) OF Redirection

Redirection ::= SEQUENCE {
        intended-recipient-name IntendedRecipientName,
        redirection-reason RedirectionReason }

IntendedRecipientName ::= SEQUENCE {
        OrAddressAndOptionalDirectoryName,
        redirection-time Time }

RedirectionReason ::= ENUMERATED {
        recipient-assigned-alternate-recipient (0),
        originator-requested-alternate-recipient (1),
        recipient-MD-assigned-alternate-recipient (2) }

dl-expansion-history EXTENSION
        DLExpansionHistory
        ::= 26

DLExpansionHistory ::= SEQUENCE SIZE (1. .ub-dl-expansions) OF DLExpansion

DLExpansion ::= SEQUENCE {
        ORAddressAndOptionalDirectoryName,
        dl-expansion-time Time }

physical-forwarding-address EXTENSION
        PhysicalForwardAddress
        ::= 27

PhysicalForwardingAddress ::= ORAddressAndOptionalDirectoryName

recipient-certificate EXTENSION
        RecipientCertificate
        ::= 28

proof-of-delivery EXTENSION
        ProofOfDelivery
        ::= 29

originator-and-DL-expansion-history EXTENSION
        OriginatorAndDLExpansionHistory
        ::= 30

OriginatorAndDLExpansionHistory ::= SEQUENCE SIZE (0. .ub-dl-expansions) OF OriginatorAndDLExpansion

OriginatorAndDLExpansion ::= SEQUENCE {
        originator-or-dl-name ORAddressAndOptionalDirectoryName,
        origination-or-expansion-time TIME }
```

FIGURE 2/X.411 (Part 26 of 41)

Abstract syntax definition of the MTS abstract service

```
reporting-DL-name EXTENSION
        ReportingDLName
        ::= 31

ReportingDLName ::= ORAddressAndOptionalDirectoryName

reporting-MTA-certificate EXTENSION
        ReportingMTACertificate
        CRITICAL FOR DELIVERY
        ::= 32

ReportingMTACertificate ::= Certificates

report-origin-authentication-check EXTENSION
        ReportOriginAuthenticationCheck
        CRITICAL FOR DELIVERY
        ::= 33

ReportOriginAuthenticationCheck ::= SIGNATURE SEQUENCE {
        algorithm-identifier ReportOriginAuthenticationAlgorithmIdentifier,
        content-identifier ContentIdentifier OPTIONAL,
        message-security-label MessageSecurityLabel OPTIONAL,
        per-recipient SEQUENCE SIZE (1. .ub-recipients) OF PerRecipientReportFields }

ReportOriginAuthenticationAlgorithmIdentifier ::= AlgorithmIdentifier

PerRecipientReportFields ::= SEQUENCE {
        actual-recipient-name ActualRecipientName,
        originally-intended-recipient-name OriginallyIntendedRecipientName OPTIONAL,
        CHOICE {
                delivery [0] PerRecipientDeliveryReportFields,
                non-delivery [1] PerRecipientNonDeliveryReportFields } }

PerRecipientDeliveryReportFields ::= SEQUENCE {
        message-delivery-time MessageDeliveryTime,
        type-of-MTS-user TypeOfMTSUser,
        recipient-certificate [0] RecipientCertificate OPTIONAL,
        proof-of-delivery [1] ProofOfDelivery OPTIONAL }

PerRecipientNonDeliveryReportFields ::= SEQUENCE {
        non-delivery-reason-code NonDeliveryReasonCode,
        non-delivery-diagnostic-code NonDeliveryDiagnosticCode OPTIONAL }
```

FIGURE 2/X.411 (Part 27 of 41)

Abstract syntax definition of the MTS abstract service

```
originating-MTA-certificate EXTENSION
        OriginatingMTACertificate
        ::= 34

OriginatingMTACertificate ::= Certificates

proof-of-submission EXTENSION
        ProofOfSubmission
        ::= 35

ProofOfSubmission ::= SIGNATURE SEQUENCE {
        algorithm-identifier ProofOfSubmissionAlgorithmIdentifier,
        message-submission-envelope MessageSubmissionEnvelope,
        content Content,
        message-submission-identifier MessageSubmissionIdentifier,
        message-submission-time MessageSubmissionTime }

ProofOfSubmissionAlgorithmIdentifier ::= AlgorithmIdentifier
```

FIGURE 2/X.411 (Part 28 of 41)

Abstract syntax definition of the MTS abstract service

```
-- Common parameter types

Content ::= OCTET STRING        -- when the content-type has the integer value external,
                                -- the value of the content octet string is the ASN.1
                                -- encoding of the external content an external-content
                                -- is a data type EXTERNAL

MTSIdentifier ::= [APPLICATION 4] SEQUENCE {
        global-domain-identifier GlobalDomainIdentifier,
        local-identifier LocalIdentifier }

LocalIdentifier ::= IA5String (SIZE (1. .ub-local-id-length))

GlobalDomainIdentifier ::= [APPLICATION 3] SEQUENCE {
        country-name CountryName,
        administration-domain-name AdministrationDomainName,
        private-domain-identifier PrivateDomainIdentifier OPTIONAL }

PrivateDomainIdentifier ::= CHOICE {
        numeric NumericString (SIZE (1. .ub-domain-name-length)),
        printable PrintableString (SIZE (1. .ub-domain-name-length)) }

MTAName ::= IA5String (SIZE (1. .ub-mta-name-length))

Time ::= UTCTime
```

FIGURE 2/X.411 (Part 29 of 41)

Abstract syntax definition of the MTS abstract service

```
-- O/R names

ORAddressAndOrDirectoryName ::= ORName

ORAddressAndOptionalDirectoryName ::= ORName

ORName ::= [APPLICATION 0] SEQUENCE {
        address COMPONENTS OF ORAddress,
        directory-name [0] Name OPTIONAL }

ORAddress ::= SEQUENCE {
        standard-attributes StandardAttributes,
        domain-defined-attributes DomainDefinedAttributes OPTIONAL,
                -- also see teletex-domain-defined-attributes
        extension-attributes ExtensionAttributes OPTIONAL }

-- Note – The OR-address is semantically absent from the OR-name if the standard-attribute sequence is empty
-- and the domain-defined-attributes and extension-attributes are both omitted.


-- Standard attributes

StandardAttributes ::= SEQUENCE
        country-name CountryName OPTIONAL,
        administration-domain-name AdministrationDomainName OPTIONAL,
        network-address [0] NetworkAddress OPTIONAL, -- also see extended-network-address
        terminal-identifier [1] TerminalIdentifier OPTIONAL,
        private-domain-name [2] PrivateDomainName OPTIONAL,
        organization-name [3] OrganizationName OPTIONAL, --also see teletex-organization-name
        numeric-user-identifier [4] NumericUserIdentifier OPTIONAL,
        personal-name [5] PersonalName OPTIONAL, -- also see teletex-personal-name
        organizational-unit-names [6] OrganizationalUnitNames OPTIONAL
                -- also see teletex-organizational-unit-names -- }

CountryName ::= [APPLICATION 1] CHOICE {
        x121-dcc-code NumericString (SIZE (ub-country-name-numeric-length)),
        iso-3166-alpha2-code PrintableString (SIZE (ub-country-name-alpha-length)) }

AdministrationDomainName ::= [APPLICATION 2] CHOICE {
        numeric NumericString (SIZE (0. .ub-domain-name-length)),
        printable PrintableString (SIZE (0. .ub-domain-name-length)) }

NetworkAddress ::= x121Address

X121Address ::= NumericString (SIZE (1. .ub-x121-address-length))

TerminalIdentifier ::= PrintableString (SIZE (1. .ub-terminal-id-length))
```

FIGURE 2/X.411 (Part 30 of 41)

Abstract syntax definition of the MTS abstract service

```
PrivateDomainName ::= CHOICE {
        numeric NumericString (SIZE (1. .ub-domain-name-length)),
        printable PrintableString (SIZE (1. .ub-domain-name-length)) }

OrganizationName ::= Printable String (SIZE (1. .ub-organization-name-length))

NumericUserIdentifier ::= NumericString (SIZE (1. .ub-numeric-user-id-length))

Personal Name ::= SET {
        surname [0] PrintableString (SIZE (1. .ub-surname-length)),
        given-name [1] PrintableString (SIZE (1. .ub-given-name-length)) OPTIONAL,
        initials [2] PrintableString (SIZE (1. .ub-initials-length)) OPTIONAL,
        generation-qualifier [3] PrintableString (SIZE (1. .ub-generation-qualifier-length)) OPTIONAL }

OrganizationUnitNames ::= SEQUENCE SIZE (1. .ub-organizational-units) OF OrganizationUnitName

OrganizationUnitName ::= PrintableString (SIZE (1. .ub-organizational-unit-name-length))


-- Domain-defined attributes

DomainDefinedAttributes ::= SEQUENCE SIZE (1. .ub-domain-defined-attributes) OF DomainDefinedAttribute

DomainDefinedAttribute ::= SEQUENCE {
        type PrintableString (SIZE (1. .ub-domain-attribute-type-length)),
        value PrintableString (SIZE (1. .ub-domain-defined-attribut-value-length)) }


-- Extension attributes

ExtensionAttributes ::= SET SIZE (1. .ub-extension-attributes) OF ExtensionAttribute

ExtensionAttribute ::= SEQUENCE {
        extension-attribute-type [0] EXTENSION-ATTRIBUTE,
        extension-attribute-value [1] ANY DEFINED BY extension-attribute-type }

EXTENSION-ATTRIBUTE MACRO ::=
BEGIN

TYPE NOTATION::= TYPE | empty
VALUE NOTATION ::= value (VALUE INTEGER (0. .ub-extension-attributes))

END  -- of EXTENTION-ATTRIBUTE
```

FIGURE 2/X.411 (Part 31 of 41)

Abstract syntax definition of the MTS abstract service

```
common-name EXTENSION-ATTRIBUTE
        CommonName
        ::= 1

CommonName ::= PrintableString (SIZE (1...ub-common-name-length))

teletex-common-name EXTENSION-ATTRIBUTE
        TeletexCommonName
        ::= 2

TeletexCommonName ::= TeletexString (SIZE (1...ub-common-name-length))

teletex-organization-name EXTENSION-ATTRIBUTE
        TeletexOrganizationalName
        ::= 3

TeletexOrganizationalName ::= TeletexString (SIZE (1...ub-organization-name-length))

teletex-personal-name EXTENSION-ATTRIBUTE
        TeletexPersonalName
        ::= 4

TeletexPersonalName ::= SET{
        surname [0] TeletexString (SIZE (1...ub-surname-length)),
        given-name [1] TeletexString (SIZE (1...ub-given-name-length)) OPTIONAL,
        initials [2] TeletexString (SIZE (1...ub-initials-length)) OPTIONAL,
        generation-qualifier [3] TeletexString (SIZE (1...ub-generation-qualifier-length)) OPTIONAL}

teletex-organizational-unit-names EXTENSION-ATTRIBUTE
        TeletexOrganizationUnitNames
        ::= 5

TeletexOrganizationUnitNames ::= SEQUENCE (SIZE (1...ub-organizational-units) OF
        TeletexOrganizationalUnitName

TeletexOrganizationalUnitName ::= TeletexString (SIZE (1...ub-organizational-unit-name-length))

teletex-domain-defined-attributes EXTENSION-ATTRIBUTE
        TeletexDomainDefinedAttributes
        ::= 6

TeletexDomainDefinedAttributes ::= SEQUENCE SIZE (1...ub-domain-defined-attributes) OF
        TeletexDomainDefinedAttribute
```

FIGURE 2/X.411 (Part 32 of 41)

Abstract syntax definition of the MTS abstract service

```
TeletexDomainDefinedAttribute ::= SEQUENCE {
        typeTeletexString (SIZE (1. .ub-domain-defined-attribute-type-length)),
        value TeletexString (SIZE (1. .ub-domain-defined-attribute-value-length)) }

pds-name EXTENSION-ATTRIBUTE
        PDSName
        ::= 7

PDSName ::= PrintableString (SIZE (1. .ub-pds-name-length))

physical-delivery-country-name EXTENSION-ATTRIBUTE
        PhysicalDeliveryCountryName
        ::= 8

PhysicalDeliveryCountryName ::= CHOICE {
        x121-dcc-code NumericString (SIZE (ub-country-name-numeric-length)),
        iso-3166-alpha2-code PrintableString (SIZE (ub-country-name-alpha-length)) }

postal-code EXTENSION-ATTRIBUTE
        PostalCode
        ::= 9

PostalCode ::= CHOICE {
        numeric-code NumericString (SIZE (1. .ub-postal-code-length)),
        printable-code PrintableString (SIZE (1. .ub-postal-code-length)) }

physical-delivery-office-name EXTENSION-ATTRIBUTE
        PhysicalDeliveryOfficeName
        ::= 10

PhysicalDeliveryOfficeName ::= PDS Parameter

physical-delivery-office-number EXTENSION-ATTRIBUTE
        PhysicalDeliveryOfficeNumber
        ::= 11

PhysicalDeliveryOfficeNumber ::= PDS Parameter

extension-OR-address-components EXTENSION-ATTRIBUTE
        ExtensionORAddressComponents
        ::= 12
```

FIGURE 2/X.411 (Part 33 of 41)

Abstract syntax definition of the MTS abstract service

```
ExtensionORAddressComponents ::= PDS Parameter

physical-delivery-personal-name EXTENSION-ATTRIBUTE
        PhysicalDeliveryPersonalName
        ::= 13

PhysicalDeliveryPersonalName ::= PDS Parameter

physical-delivery-organization-name EXTENSION-ATTRIBUTE
        PhysicalDeliveryOrganizationName
        ::= 14

PhysicalDeliveryOrganizationName ::= PDS Parameter

extension-physical-delivery-address-components EXTENSION-ATTRIBUTE
        ExtensionPhysicalDeliveryAddressComponents
        ::= 15

ExtensionPhysicalDeliveryAddressComponents ::= PDS Parameter

unformatted-postal-address EXTENSION-ATTRIBUTE
        UnformattedPostalAddress
        ::= 16

UnformattedPostalAddress ::= SET{
        printable-address SEQUENCE SIZE (1. .ub-pds-physical-address-lines) OF
                PrintableString (SIZE (1. .ub-pds-parameter-length)) OPTIONAL,
        teletex-string TeletexString (SIZE (1. .ub-unformatted-address-length)) OPTIONAL}

street-address EXTENSION-ATTRIBUTE
        StreetAddress
        ::= 17

StreetAddress ::= PDS Parameter
```

FIGURE 2/X.411 (Part 34 of 41)

Abstract syntax definition of the MTS abstract service

```
post-office-box-address EXTENSION-ATTRIBUTE
        PostOfficeBoxAddress
        ::= 18

PostOfficeBoxAddress ::= PDS Parameter

poste-restante-address EXTENSION-ATTRIBUTE
        PosteRestanteAddress
        ::= 19

PosteRestanteAddress ::= PDS Parameter

unique-postal-name EXTENSION-ATTRIBUTE
        UniquePostalName
        ::= 20

UniquePostalName ::= PDS Parameter

local-postal-attributes EXTENSION-ATTRIBUTE
        LocalPostalAttributes
        ::= 21

LocalPostalAttributes ::= PDS Parameter

PDS Parameter ::= SET
        printable-string PrintableString (SIZE (1. . .ub-pds-parameter-length)) OPTIONAL,
        teletex-string TeletexString (SIZE (1. . .ub-pds-parameter-length)) OPTIONAL }

extended-network-address EXTENSION-ATTRIBUTE
        ExtendedNetworkAddress
        ::= 22

ExtendedNetworkAddress ::= CHOICE {
        e163-4-address SEQUENCE {
                number [0] NumericString (SIZE (1. .ub-e163-4-number-length)),
                sub-address [1] NumericString (SIZE (1. .ub-e163-4-sub-address-length)) OPTIONAL },
        psap-address [0] PresentationAddress }

terminal-type EXTENSION-ATTRIBUTE
        TerminalType
        ::= 23
```

FIGURE 2/X.411 (Part 35 of 41)

Abstract syntax definition of the MTS abstract service

```
TerminalType ::= INTEGER {
        telex (3),
        teletex (4),
        g3-facsimile (5),
        g4-facsimile (6),
        ia5-terminal (7),
        videotex (8) } (0. .ub-integer-options)
```

FIGURE 2/X.411 (Part 36 of 41)

Abstract syntax definition of the MTS abstract service

```
-- Encoded information types

EncodedInformationTypes ::= [APPLICATION 5] SET{
        built-in-encoded-information-types [0] BuiltInEncodedInformationTypes,
        non-basic-parameters COMPONENTS OF NonBasicParameters,
        external-encoded-information-types [4] ExternalEncodedInformationTypes OPTIONAL}


-- Built-in encoded information types

Built-inEncodedInformationTypes ::= BIT STRING{
        undefined (0),
        telex (1),
        ia5-text (2),
        g3-facsimile (3),
        g4-class-1 (4),
        teletex (5),
        videotex (6),
        voice (7),
        sfd (8),
        mixed-mode (9)} (SIZE (0. .ub-built-i n-encoded-information-types))


-- Non-basic parameters

NonBasicParameters ::= SET{
        g3-facsimile [1] G3FacsimileNonBasicParameters DEFAULT{},
        teletex [2] TeletexNonBasicParameters DEFAULT{},
        g4-class-1-and-mixed-mode [3] G4Class1AndMixedModeNonBasicParameters OPTIONAL}

G3FacsimileNonBasicParameters ::= BIT STRING{
        two-dimensional (8),
        fine-resolution (9),
        unlimited-length (20),
        b4-length (21),
        a3-width (22),
        b4-width (23),
        uncompressed (30)}                          -- as defined in Recommendation T.30

TeletexNonBasicParameters ::= SET{
        graphic-character-sets [0] TeletexString OPTIONAL,
        control-character-sets [1] TeletexString OPTIONAL,
        page-formats [2] OCTET STRING OPTIONAL,
        miscellaneous-terminal-capabilities [3] TeletexString OPTIONAL,
        private-use [4] OCTET STRING OPTIONAL       -- maximum ub-teletex-private-use-length octets --}
        -- as defined in Recommendation T.62
```

FIGURE 2/X.411 (Part 37 of 41)

Abstract syntax definition of the MTS abstract service

```
G4Class1AndMixedModeNonBasicParameters ::= PresentationCapabilities

PresentationCapabilities ::= ANY        -- as defined in Recommendations T.400, T.503 and T.501


-- External encoded information types

ExternalEncodedInformationTypes ::= SET SIZE (1. .ub-encoded-information-types) OF
        ExternalEncodedInformationType

ExternalEncodedInformationType ::= OBJECT IDENTIFIER
```

FIGURE 2/X.411 (Part 38 of 41)

Abstract syntax definition of the MTS abstract service

```
-- Token

Token ::= SEQUENCE {
        token-type-identifier [0] TOKEN,
        token [1] ANY DEFINED BY token-type-identifier }

TOKEN MACRO ::=
BEGIN

TYPE NOTATION ::= type | empty
VALUE NOTATION ::= (VALUE OBJECT IDENTIFIER)

END   -- of TOKEN

asymmetric-token TOKEN
                AsymmetricToken
        ::= id-tok-asymmetricToken

AsymmetricToken ::= SIGNED SEQUENCE {
        signature-algorithm-identifier AlgorithmIdentifier,
        recipient-name RecipientName,
        time Time,
        signed-data [0] TokenData OPTIONAL,
        encryption-algorithm-identifier [1] AlgorithmIdentifier OPTIONAL,
        encrypted-data [2] ENCRYPTED TokenData OPTIONAL }

TokenData ::= SEQUENCE {
        type [0] TOKEN-DATA,
        value [1] ANY DEFINED BY type }

TOKEN-DATA MACRO ::=
BEGIN

TYPE NOTATION ::= type | empty
VALUE NOTATION ::= value (VALUE INTEGER)

END   -- TOKEN-DATA

bind-token-signed-data TOKEN-DATA
                BindTokenSignedData
        ::=

BindTokenSignedData ::= RandomNumber
```

FIGURE 2/X.411 (Part 39 of 41)

Abstract syntax definition of the MTS abstract service

```
RandomNumber ::= BIT STRING

message-token-signed-data TOKEN-DATA
                MessageTokenSignedData
        ::= 2

MessageTokenSignedData ::= SEQUENCE{
        content-confidentiality-alogorithm-identifier [0] ContentConfidentialityAlgorithmIdentifier
                OPTIONAL,
        content-integrity-check [1] ContentIntegrityCheck OPTIONAL,
        message-security-label [2] MessageSecurityLabel OPTIONAL,
        proof-of-delivery-request [3] ProofOfDeliveryRequest OPTIONAL,
        message-sequence-number [4] INTEGER OPTIONAL}

message-token-encrypted-data TOKEN-DATA
                MessageTokenEncryptedData
        ::= 3

MessageTokenEncryptedData ::= SEQUENCE{
        content-confidentiality-key [0] EncryptionKey OPTIONAL,
        content-integrity-check [1] ContentIntegrityCheck OPTIONAL,
        message-security-label [2] MessageSecurityLabel OPTIONAL,
        content-integrity-key [3] EncryptionKey OPTIONAL,
        message-sequence-number [4] INTEGER OPTIONAL}

EncryptionKey ::= BIT STRING


-- Security label

Security label ::= SET{
        security-policy-identifier SecurityPolicyIdentifier OPTIONAL,
        security-classification SecurityClassification OPTIONAL,
        privacy-mark PrivacyMark OPTIONAL,
        security-categories SecurityCategories OPTIONAL}

SecurityPolicyIdentifier ::= OBJECT IDENTIFIER

SecurityClassification ::= INTEGER{
        unmarked (0),
        unclassified (1),
        restricted (2),
        confidential (3),
        secret (4),
        top-secret (5)} (0. . .ub-integer-options)
```

FIGURE 2/X.411 (Part 40 of 41)

Abstract syntax definition of the MTS Abstract Service

```
PrivacyMark ::= PrintableString (SIZE (1. .ub-privacy-mark-length))

SecurityCategories ::= SET SIZE (1. .ub-security-categories) OF SecurityCategory

SecurityCategory ::= SEQUENCE {
        type [0] SECURITY-CATEGORY,
        value [1] ANY DEFINED BY type }

SECURITY-CATEGORY MACRO ::=
BEGIN

TYPE NOTATION ::= type | empty
VALUE NOTATION ::= value (VALUE OBJECT IDENTIFIER)

END  -- of SECURITY-CATEGORY

END  -- of MTSAbstractService
```

FIGURE 2/X.411 (Part 41 of 41)

Abstract syntax definition of the MTS abstract service

10 Refined message transfer system model

Paragraph 6 describes the MTS as an object, without reference to its internal structure. This paragraph refines the MTS model, and exposes its component objects and the ports shared between them.

Figure 3/X.411 models the MTS and reveals its internal structure.

The MTS comprises a collection of message-transfer-agent (MTA) objects, which cooperate together to form the MTS and offer the MTS abstract service to its users. It is the MTAs which perform the active functions of the MTS, i.e. transfer of messages, probes and reports, generation of reports, and content conversion.

MTA objects also have ports, some of which are precisely those which are also visible at the boundary of the MTS object, i.e. submission-ports, delivery-ports and administration ports. However, MTAs also have another type of port – which are concerned with the distribution of the MTS abstract service between the MTAs, and are not visible at the boundary of the MTS object.

A transfer-port enables an MTA to transfer messages, probes and reports to another MTA. In general, a message, probe or report may have to be transferred a number of times between different MTAs to reach its intended destination.

If a message is addressed to multiple recipients served by several different MTAs, the message must be transferred through the MTS along several different paths. From the perspective of an MTA transferring such a message, some recipients may be reached via one path while other recipients may be reached via another. At such an MTA, two copies of the message are created, and each is transferred to the next MTA along its respective path. The copying and branching of the message is repeated until each copy has reached a final destination MTA, where the message can be delivered to one or more recipient MTS-users.

Every MTA along a path taken by a message is responsible for delivering or transferring the message to a particular subset of the originally-specified-recipients. Other MTAs take care of the deliver or transfer to remaining recipients, using copies of the messages created along the way.

Reports on the delivery or non-delivery of a message to one or more recipient MTS-users, are generated by MTAs in accordance with the request of the originator of the message and the originating-MTA. An MTA may generate a delivery-report upon successfully delivering a copy of a message to a recipient MTS-user. It may generate a non-delivery-report upon determining that a copy of a message is undeliverable to one or more recipients, that is, it is unable to deliver the message to the recipient MTS-users, or it is unable to transfer the message to an adjacent MTA that would take responsibility for delivery or transferring the message further.

For efficiency, an MTA may generate a single, combined report that applies to several copies of a single, multiple recipient message for which it is responsible. Both delivery- and non-delivery-reports may be combined together. However, in order for reports to be combined in this manner, the same content conversion, if any, must have been performed on the message for all recipients to whom the report refers.

Reports that pertain to copies of the same multiple recipient message but that were generated by different MTAs are not combined by any intermediate MTAs, but instead remain distinct.

When required, an MTA may perform content conversion. When neither the originating nor the recipient MTS-user requests nor prohibits conversion, implicit conversion of a message's encoded-information-types may be performed by an MTA to suit the encoded-information-types that the recipient MTS-user is able to receive. The originating MTS-user may also explicitly request conversion of specific encoded-information-types for a particular recipient MTS-user.

The submission-, delivery- and administration-ports of an MTA, which are also visible at the boundary of the MTS, are defined in Section 2 of this Recommendation. The remaining paragraphs in this section define the transfer-port of an MTA, and the procedures performed by MTAs to ensure the correct distributed operation of the MTS.

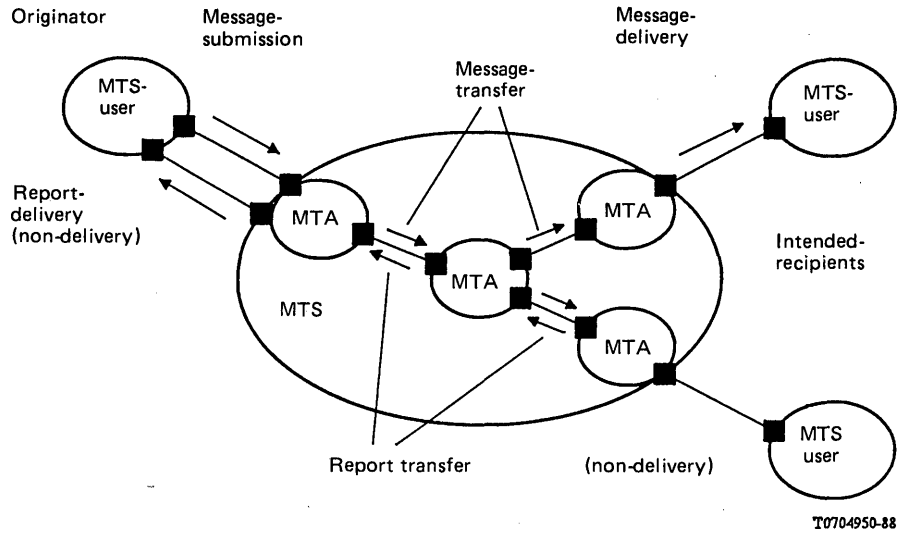


FIGURE 3/X.411

Refined message transfer system model

11 Message transfer agent abstract service overview

Section two defines the MTS abstract service provided by the submission-, delivery- and administration-ports of an MTA. This paragraph defines the following abstract-operations that are provided by the transfer-ports of MTAs:

MTA-bind and MTA-unbind

a) MTA-bind

b) MTA-unbind.

Transfer port abstract-operations

c) message-transfer

d) probe-transfer

e) report-transfer.

11.1 *MTA-bind and MTA-unbind*

The **MTA-bind** enables an MTA to establish an association with another MTA. Abstract-operations other than MTA-bind can only be invoked in the context of an established association.

The **MTA-unbind** enables an MTA to establish an association with another MTA. Abstract-operations other than MTA-bind can only be invoked in the context of an established association.

11.2 *Transfer port abstract-operations*

The **message-transfer** abstract-operation enables an MTA to transfer a message to another MTA.

The **probe-transfer** abstract-operation enables an MTA to transfer a probe to another MTA.

The **report-transfer** abstract-operation enables an MTA to transfer a report to another MTA.

12 Message transfer agent abstract service definition

The MTS abstract service is defined in § 8. This paragraph defines the semantics of the parameters of the abract-service provided by the transfer-port of MTAs.

Paragraph 12.1 defines the MTA-bind and MTA-unbind. Paragraph 12.2 defines the transfer-port. Paragraph 12.3 defines some common parameter types.

The abstract-syntax of the MTA abstract service is defined in § 13.

12.1 *MTA-bind and MTA-unbind*

This paragraph defines the abstract-service used to establish and release associations between MTAs.

12.1.1 *Abstract-bind and abstract-unbind*

This paragraph defines the following abstract-bind and abstract-unbind:

a) MTA-bind

b) MTA-unbind.

12.1.1.1 *MTA-bind*

The MTA-bind enables an MTA to establish an association with another MTA.

The MTA-bind establishes the **credentials** of MTAs to interact, and the **application-context** and **security-context** of the association. An assocition can only be released by the initiator of that association (using MTA-unbind).

Abstract-operations other than MTA-bind can only be invoked in the context of an established association.

The successful completion of the MTA-bind signifies the establishment of an association.

The disruption of the MTA-bind by a bind-error indicates that an association has not been established.

12.1.1.1.1 *Arguments*

Table 27/X.411 lists the arguments of the MTA-bind, and for each argument qualifies its presence and indicates the paragraph in which the argument is defined.

TABLE 27/X.411

MTA-bind arguments

Argument	Presence	Clause
Bind arguments		
Initiator-name	O	12.1.1.1.1.1
Initiator-credentials	O	12.1.1.1.1.2
Security-context	O	12.1.1.1.1.3

12.1.1.1.1.1 *Initiator-name*

This argument contains a name for the initiator of the association. It may be generated by the initiator of the association.

The name of an **MTA-name**.

12.1.1.1.1.2 *Initiator-credentials*

This argument contains the **credentials** of the initiator of the association. It may be generated by the initiator of the association.

The **initiator-credentials** may be used by the responder to authenticate the identity of the initiator (see Recommendation X.509).

If only simple-authentication is proposed, the **initiator-credentials** comprise a simple **password** associated with the **initiator-name**.

If strong-authentication is used, the **initiator-credentials** comprise an **initiator-bind-token** and, optionally, an **initiator-certificate**.

The **initiator-bind-token** is a **token** generated by the initiator of the association. If the **initiator-bind-token** is an **asymmetric-token**, the **signed-data** comprises a **random-number**. The **encrypted-data** of an **asymmetric-token** may be used to convey secret security-relevant information (e.g., one or more symmetric-encryption-keys) used to secure the association, or may be absent from the **initiator-bind-token**.

The **initiator-certificate** is a **certificate** of the initiator of the association, generated by a trusted source (e.g., a certification-authority). It may be supplied by the initiator of the association, if the **initiator-bind-token** is an **asymmetric-token**. The **initiator-certificate** may be used to convey a verified copy of the public-asymmetric-encryption-key (**subject-key**) of the initiator of the association. The initiator's public-asymmetric-encryption-key may be used by the responder to compute the **responder-bind-token**. If the responder is known to have, or have access to, the initiator's certificate (e.g., via the Directory), the **initiator-certificate** may be omitted.

12.1.1.1.1.3 *Security-context*

This argument indicates the **security-context** that the initiator of the association proposes to operate at. It may be generated by the initiator of the association.

The **security-context** comprises one or more **security-labels** that defines the sensitivity of interactions that may occur between the MTAs for the duration of the association, in line with the security-policy in force. The **security-context** shall be one that is allowed by the **security-labels** associated with the MDs (MTAs).

If **security-contexts** are not established between the MTAs, the sensitivity of interactions that may occur between the MTAs may be at the discretion of the invoker of an abstract-operation.

12.1.1.1.2 *Results*

Table 28/X.411 lists the results of the MTA-bind, and for each result qualifies its presence and indicates the paragraph in which the result is defined.

TABLE 28/X.411

MTA-bind results

Result	Presence	Clause
Bind results		
Responder-name	O	12.1.1.1.2.1
Responder-credentials	O	12.1.1.1.2.2

12.1.1.1.2.1 *Responder-name*

This argument contains a name for the responder of the association. It may be generated by the responder of the association.

The name is an **MTA-name**.

12.1.1.1.2.2 *Responder-credentials*

This argument contains the **credentials** of the responder of the association. It may be generated by the responder of the association.

The **responder-credentials** may be used by the initiator to authenticate the identity of the responder (see Recommendation X.509).

If only simple-authentication is used, the **responder-credentials** comprise a simple **password** associated with the **responder-name**.

If strong-authentication is used, the **responder-credentials** comprise a **responder-bind-token**. The **responder-bind-token** is a **token** generated by the responder of the association. The **responder-bind-token** shall be the same type of **token** as the **initiator-bind-token**. If the **responder-bind-token** is an **asymmetric-token**, the **signed-data** comprises a **random-number** (which may be related to the **random-number** supplied in the **initiator-bind-token**). The **encrypted-data** of an **asymmetric-token** may be used to convey security-relevant information (e.g., one or more symmetric-encryption-keys) used to secure the association, or may be absent from the **responder-bind-token**.

12.1.1.1.3 *Bind-errors*

The bind-errors that may disrupt the MTA-bind are defined in § 12.1.2.

12.1.1.2 *MTA-unbind*

The MTA-unbind enables the release of an established association by the initiator of the association.

12.1.1.2.1 *Arguments*

The MTA-unbind service has no arguments.

12.1.1.2.2 *Results*

The MTA-unbind service returns an empty result as indication of release of the association.

12.1.1.2.3 *Unbind-errors*

There are no unbind-errors that may disrupt the MTA-unbind.

12.1.2 *Bind-errors*

This paragraph defines the following bind-errors:

a) authentication-error,
b) busy,
c) unacceptable-dialogue-mode,
d) unacceptable-security-context.

12.1.2.1 *Authentication-error*

The authentication-error bind-error reports that an association cannot be established due to an authentication error; the initiator's **credentials** are not acceptable or are improperly specified.

The authentication-error bind-error has no parameters.

12.1.2.2 *Busy*

The busy bind-error reports that an association cannot be established because the responder is busy.

The busy bind-error has no parameters.

12.1.2.3 *Unacceptable-dialogue-mode*

The unacceptable-dialogue-mode bind-error reports that the dialogue-mode proposed by the initiator of the association is unacceptable to the responder (see § 12 of Recommendation X.419).

The unacceptable-dialogue-mode bind-error has no parameters.

12.1.2.4 *Unacceptable-security-context*

The unacceptable-security-context-bind-error reports that the **security-context** proposed by the initiator of the association is unacceptable to the responder.

The Unacceptable-security-context bind-error has no parameters.

12.2 *Transfer port*

This paragraph defines the abstract-operations and abstract-errors which occur at a transfer-port.

12.2.1 *Abstract-operations*

This paragraph defines the following transfer-port abstract-operations:

a) message-transfer,
b) probe-transfer,
c) report-transfer.

12.2.1.1 *Message-transfer*

The message-transfer abstract-operation enables the MTA to transfer a message to another MTA.

12.2.1.1.1 *Arguments*

Table 29/X.411 lists the arguments of the message-transfer abstract-operation, and for each argument qualifies its presence and identifies the paragraph in which the argument is defined.

12.2.1.1.1.1 *Message-identifier*

This argument contains an **MTS-identifier** that distinguishes the message from all other messages, probes and reports within the MTS. It shall be generated by the originating-MTA of the message, and shall have the same value as the **message-submission-identifier** supplied to the originator of the message when the message was submitted, and the **message-delivery-identifier** supplied to the recipient of the message when the message is delivered.

When a message is copied for routing to multiple recipients via different MTAs, each copy of the message bears the **message-identifier** of the original. The copies can be distinguished from one another by the **originally-specified-recipient-number** and the corresponding **responsibility** arguments, which specify to which recipient(s) each copy is to be delivered.

12.2.1.1.1.2 *Per-domain-bilateral-information*

This argument contains information intended for MDs which the message will encounter as it is transferred through the MTS. It may be generated by the originating-MD of the message.

This argument may contain zero or more elements, each of which comprises:

– the **bilateral-information** intended for an MD;
– the **country-name**, the **administration-domain-name** and, optionally, the **private-domain-identifier** of the MD for which the **bilateral-information** is intended.

12.2.1.1.1.3 *Trace-information*

This argument documents the actions taken on the message (or probe or report) by each MD through which the message (or probe or report) passes as it is transferred through the MTS (see § 12.3.1). It shall be generated by each MD through which the message (or probe or report) passes.

12.2.1.1.1.4 *Internal-trace-information*

This argument documents the actions taken on the message (or probe or report) by each MTA through which the message (or probe or report) passes as it is transferred within an MD (see § 12.3.1). It shall be generated by each MTA through which the message (or probe or report) passes within an MD.

This argument shall not be supplied by the invoker of the message-transfer abstract-operation when transferring a message to another MD, unless by bilateral agreement between MDs.

12.2.1.1.1.5 *Originally-specified-recipient-number*

This argument, combined with the **message-identifier**, unambiguously identifies the copy of the message delivered to each recipient. It shall be generated by the originating-MTA of the message. A different value of this argument is specified for each recipient of the message.

The **originally-specified-recipient-number** is an integer value in the range that begins with one and ends with the number of originally-specified-recipients.

TABLE 29/X.411

Message-transfer arguments

Argument	Presence	Clause
Relaying arguments		
Message-identifier	M	12.2.1.1.1.1
Per-domain-bilateral-information	C	12.2.1.1.1.2
Trace-information	M	12.2.1.1.1.3
Internal-trace-information	C	12.2.1.1.1.4
DL-expansion-history	C	8.3.1.1.1.7
Originator argument		
Originator-name	M	8.2.1.1.1.1
Recipient arguments		
Recipient-name	M	8.2.1.1.1.2
Originally-specified-recipient-number	M	12.2.1.1.1.5
Responsibility	M	12.2.1.1.1.6
DL-expansion-prohibited	C	8.2.1.1.1.6
Disclosure-of-recipients	C	8.2.1.1.1.7
Redirection arguments		
Alternate-recipient-allowed	C	8.2.1.1.1.3
Recipient-reassignment-prohibited	C	8.2.1.1.1.4
Originator-requested-alternate-recipient	C	8.2.1.1.1.5
Intended-recipient-name	C	8.3.1.1.1.4
Redirection-reason	C	8.3.1.1.1.5
Priority argument		
Priority	C	8.2.1.1.1.8
Conversion arguments		
Implicit-conversion-prohibited	C	8.2.1.1.1.9
Conversion-with-loss-prohibited	C	8.2.1.1.1.10
Explicit-conversion	C	8.2.1.1.1.11
Delivery time arguments		
Deferred-delivery-time	C	12.2.1.1.1.7
Latest-delivery-time	C	8.2.1.1.1.13
Delivery method argument		
Requested-delivery-method	C	8.2.1.1.1.14
Physical delivery arguments		
Physical-forwarding-prohibited	C	8.2.1.1.1.15
Physical-forwarding-address-request	C	8.2.1.1.1.16
Physical-delivery-modes	C	8.2.1.1.1.17
Registred-mail-type	C	8.2.1.1.1.18
Recipient-number-for-advice	C	8.2.1.1.1.19
Physical-rendition-attributes	C	8.2.1.1.1.20
Originator-return-address	C	8.2.1.1.1.21
Delivery report request arguments		
Originator-report-request	M	8.2.1.1.1.22
Originating-MTA-report-request	M	12.2.1.1.1.8
Content-return-request	C	8.2.1.1.1.23
Physical-delivery-report-request	C	8.2.1.1.1.24
Security arguments		
Originator-certificate	C	8.2.1.1.1.25
Message-token	C	8.2.1.1.1.26
Content-confidentiality-algorithm-identifier	C	8.2.1.1.1.27
Content-integrity-check	C	8.2.1.1.1.28
Message-origin-authentication-check	C	8.2.1.1.1.29
Message-security-label	C	8.2.1.1.1.30
Proof-of-delivery-request	C	8.2.1.1.1.32
Content arguments		
Original-encoded-information-types	C	8.2.1.1.1.33
Content-type	M	8.2.1.1.1.34
Content-identifier	C	8.2.1.1.1.35
Content-correlator	C	8.2.1.1.1.36
Content	M	8.2.1.1.1.37

There is a one-to-one relationship between a particular **originally-specified-recipient-number** value and a particular **recipient-name** at the time of message-submission; it should not be assumed that this is a singular relationship at the time of message-delivery. That is, an **originally-specified-recipient-number** value can be used to distinguish an originally specified **recipient-name**, but not an actual recipient that will receive the message.

12.2.1.1.1.6 *Responsibility*

This argument indicates whether the receiving-MTA shall have the responsibility to either deliver the message to a recipient or to transfer it to another MTA for subsequent delivery to the recipient. It shall be generated by the sending-MTA. A different value of this argument may be specified for each recipient of the message.

This argument may have one of the following values: **responsible** or **not-responsible**.

12.2.1.1.1.7 *Deferred-delivery-time*

This argument is defined in § 8.2.1.1.1.12. It may appear in a message at a transfer-port if there is a bilateral agreement that an MTA other than the originating-MTA of the message will defer the delivery of the message.

12.2.1.1.1.8 *Originating-MTA-report-request*

This argument indicates the kind of report requested by the originating-MTA. It shall be generated by the originating-MTA of the message. A different value of this argument may be specified for each recipient of the message.

This argument may have one of the following values:

- **non-delivery-report**: a report is returned only in case of non-delivery, and it contains only the **last-trace-information**;
- **report**: a report is returned in case of delivery or non-delivery, and it contains only the **last-trace-information**;
- **audited-report**: a report is returned in case of delivery or non-delivery, and it contains all of the **trace-information**.

The **originating-MTA-report-request** argument shall specify at least the report level specified in the **originator-report-request** argument, where the increasing order or report levels is **no-report, non-delivery-report, report, audited-report**.

12.2.1.1.2 *Results*

The message-transfer abstract-operation does not return a result.

12.2.1.1.3 *Abstract-errors*

There are no abstract-errors that may disrupt the message-transfer abstract-operation.

12.2.1.2 *Probe-transfer*

The probe-transfer abstract-operation enables an MTA to transfer a probe to another MTA.

12.2.1.2.1 *Arguments*

Table 30/X.411 lists the arguments of the probe-transfer abstract-operation, and for each argument qualifies its presence and identifies the paragraph in which the argument is defined.

12.2.1.2.1.1 *Probe-identifier*

This argument contains an **MTS-identifier** that distinguishes the probe from all other message, probes and reports within the MTS. It shall be generated by the originating-MTA of the probe, and shall have the same value as the **probe-submission-identifier** supplied to the originator of the probe when the probe was submitted.

12.2.1.2.2 *Results*

The probe-transfer abstract-operation does not return a result.

12.2.1.2.3 *Abstract-errors*

There are no abstract-errors that may disrupt the probe-transfer abstract-operation.

TABLE 30/X.411

Probe-transfer arguments

Argument	Presence	Clause
Relaying arguments		
Probe-identifier	M	12.2.1.2.1.1
Per-domain-bilateral-information	C	12.2.1.1.1.2
Trace-information	M	12.2.1.1.1.3
Internal-trace-information	C	12.2.1.1.1.4
DL-expansion-history	C	8.3.1.1.1.7
Originator argument		
Originator-name	M	8.2.1.1.1.1
Recipient arguments		
Recipient-name	M	8.2.1.1.1.2
Originally-specified-recipient-number	M	12.2.1.1.1.5
Responsibility	M	12.2.1.1.1.6
DL-expansion-prohibited	C	8.2.1.1.1.6
Redirection arguments		
Alternate-recipient-allowed	C	8.2.1.1.1.3
Recipient-reassignment-prohibited	C	8.2.1.1.1.4
Originator-requested-alternate-recipient	C	8.2.1.1.1.5
Intended-recipient-name	C	8.3.1.1.1.4
Redirection-reason	C	8.3.1.1.1.5
Conversion arguments		
Implicit-conversion-prohibited	C	8.2.1.1.1.9
Conversion-with-loss-prohibited	C	8.2.1.1.1.10
Explicite-conversion	C	8.2.1.1.1.11
Delivery method argument		
Request-delivery-method	C	8.2.1.1.1.14
Physical delivery argument		
Physical-rendition-attributes	C	8.2.1.1.1.20
Report request arguments		
Originator-report-request	M	8.2.1.1.1.22
Originating-MTA-report-request	M	12.2.1.1.1.8
Security arguments		
Originator-certificate	C	8.2.1.1.1.25
Probe-origin-authentication-check	C	8.2.1.2.1.1
Message-security-label	C	8.2.1.1.1.30
Content arguments		
Original-encoded-information-types	C	8.2.1.1.1.33
Content-type	M	8.2.1.1.1.34
Content-identifier	C	8.2.1.1.1.35
Content-correlator	C	8.2.1.1.1.36
Content-length	C	8.2.1.2.1.2

12.2.1.3 *Report-transfer*

The report-transfer abstract-operation enables an MTA to transfer a report to another MTA.

12.2.1.3.1 *Arguments*

Table 31/X.411 lists the arguments of the report-transfer abstract-operation, and for each argument qualifies its presence and identifies the paragraph in which the argument is defined.

12.2.1.3.1.1 *Report-identifier*

This argument contains an **MTS-identifier** that distinguishes the report from all other messages, probes and reports within the MTS. It shall be generated by the originating-MTA of the report.

12.2.1.3.1.2 *Report-destination-name*

This argument contains the **OR-name** of the immediate destination of the report. It shall be generated by the originating-MTA of the report, and subsequently modified by the DL expansion-points if any DLs had been expanded to add recipients to the subject.

The originating-MTA of the report shall set this argument to be the **originator-name** of the subject if the subject does not have a **DL-expansion-history**, or to the last **OR-name** in the **DL-expasion-history** if this is present in the subject.

A DL expansion-point may replace its own **OR-name** in this argument by the **OR-name** which immediately preceeds its own **OR-name** in the report's **originator-and-DL-expansion-history**, or some other **OR-name** according to the reporting-policy of the DL.

12.2.1.3.1.3 *Subject-identifier*

This argument contains the **message-identifier** (or **probe-identifier**) of the subject (an **MTS-identifier**). It shall be generated by the originating-MTA of the subject.

12.2.1.3.1.4 *Subject-intermediate-trace-information*

The argument contains the **trace-information** present in the subject when it was transferred into the reporting-MD. It shall be present if, and only if, an audit-and-confirmed report was requested by the originating-MTA of the subject. It may be generated by the reporting-MTA.

Note – The inclusion in the **subject-intermediate-trace-information** of the **internal-trace-information** present in the subject when it was transferred to the reporting-MTA is for further study.

12.2.1.3.1.5 *Arrival-time*

This argument contains the **time** at which the subject entered the MD making the report. It shall be generated by the originating-MD of the report. A different value of this argument may be specified for each recipient of the subject to which the report relates.

12.2.1.3.1.6 *Additional-information*

The specification of the contents of this argument is by bilateral agreement between MDs.

12.2.1.3.2 *Results*

The report-transfer abstract-operation does not return a result.

12.2.1.3.3 *Abstract-errors*

There are no abstract-errors that may disrupt the report-transfer abstract-operation.

12.2.2 *Abstract-errors*

The transfer-port has not abstract-errors.

TABLE 31/X.411

Report-transfer arguments

Argument	Presence	Clause
Relaying arguments		
Report-identifier	M	12.2.1.3.1.1
Trace-information	M	12.2.1.1.1.3
Internal-trace-information	C	12.2.1.1.1.4
Report destination argument		
Report-destination-name	M	12.2.1.3.1.2
Report request argument		
Originator-report-request	M	8.2.1.1.1.22
Subject trace arguments		
Subject-identifier	M	12.2.1.3.1.3
Originally-specified-recipient-number	M	12.2.1.1.1.5
Subject-intermediate-trace-information	C	12.2.1.3.1.4
Arrival-time	M	12.2.1.3.1.5
Originator-and-DL-expansion-history	C	8.3.1.2.1.3
Reporting-DL-name	C	8.3.1.2.1.4
Conversion argument		
Converted-encoded-information-types	C	8.3.1.2.1.5
Supplementary information arguments		
Supplementary-information	C	8.3.1.2.1.6
Physical-forwarding-address	C	8.3.1.2.1.7
Subject redirection arguments		
Actual-recipient-name	M	8.3.1.2.1.2
Intended-recipient-name	C	8.3.1.1.1.4
Redirection-reason	C	8.3.1.1.1.5
Content arguments		
Original-encoded-information-types	C	8.2.1.1.1.33
Content-type	C	8.2.1.1.1.34
Content-identifier	C	8.2.1.1.1.35
Content-correlator	C	8.2.1.1.1.36
Returned-content	C	8.3.1.2.1.14
Delivery arguments		
Message-delivery-time	C	8.2.1.2.1.8
Type-of-MTS-user	C	8.3.1.2.1.9
Non-delivery arguments		
Non-delivery-reason-code	C	8.3.1.2.1.10
Non-delivery-diagnostic-code	C	8.3.1.2.1.11
Security arguments		
Recipient-certificate	C	8.3.1.1.2.1
Proof-of-delivery	C	8.3.1.1.2.2
Reporting-MTA-certificate	C	8.3.1.2.1.12
Report-origin-authentication-check	C	8.3.1.2.1.13
Message-security-label	C	8.2.1.1.1.30
Additional information argument		
Additional-information	C	12.2.1.3.1.6

12.3 *Common parameter types*

This paragraph defines a number of common parameter types of the MTA abstract service.

12.3.1 *Trace-information and internal-trace-information*

Trace-information documents the actions taken on a message, probe or report by each MD through which it passes as it is transferred through the MTS.

Internal-trace-information documents the action taken on a message, probe or report by each TMA through which it passes as it is transferred through an MD. **Internal-trace-information** shall be removed from a message, probe or report before it is transferred out of an MD, unless by bilateral agreement between MDs.

Trace-information (or **internal-trace-information**) comprises a sequence of **trace-information-elements** (or **internal-trace-information-elements**). The first **trace-information-element** (or **internal-trace-information-element**) is that supplied by the originating-MD (or -MTA) of the message, probe or report. The second **trace-information-element** (or **internal-trace-information-element**) is that supplied by the next MD (or MTA) encountered by the message, probe or report, and so on. Each MD (or MTA) adds its **trace-information-element** (or **internal-trace-information-element**) to the end of the existing sequence. **Trace-information** is added by the first MTA encountered by the message, probe or report in each MD it passes through.

Each **trace-information-element** includes the **global-domain-identifier** of the MD supplying the **trace-information-element**.

Each **internal-trace-information-element** includes the **MTA-name** of the MTA supplying the **internal-trace-information-element** and the **global-domain-identifier** of the MD to which the MTA belongs.

Each **trace-information-element** (or **in ternal-trace-information-element**) includes the **arrival-time** at which the message, probe or report entered the MD (or MTA). In the case of the originating-MD (or -MTA) of the message, probe or report, the **arrival-time** is the time of message-submission, probe-submission or report generation, respectively.

Each **trace-information-element** (or **internal-trace-information-element**) specifies the **routing-action** the MD (or MTA) supplying the **trace-information-element** (or **internal-trace-information-element**) took with respect to the message, probe or report. **Relayed** is the normal **routing-action** of transferring the message, probe or report to another MD (or MTA). **Rerouted** indicates that an attempt had previously been made to route the message, probe or report to an **attempted-domain** (or **attempted-MTA**); the **global-domain-identifier** of the **attempted-domain** is included in the **trace-information-element**; if the rerouting attempt was to another MTA within the same MD, then the MTA-name of the **attempted-MTA** is included in the **internal-trace-information-element**; if the rerouting attempt was to another MD, then the **global-domain-identifier** of the **attempted-domain** is included in the **internal-trace-information-element** instead of an **MTA-name**.

Each **trace-information-element** (or **internal-trace-information-element**) also specifies any **additional-actions** the MD (or MTA) supplying the **trace-information-element** (or **internal-trace-information-element**) took with respect to the message, probe or report. Indications of any such **additional-actions** which appear in the **internal-trace-information-elements** during a traversal of an MD shall also be reflected in the corresponding **trace-information-element(s)** for the traversal of the MD.

If the deferred-delivery caused the MD (or MTA) supplying the **trace-information-element** (or **internal-trace-information-element**) to hold the message for a period of time, the **deferred-time** when it started to process the message for delivery or transfer is also included in the **trace-information-element** (or **internal-trace-information-element**). This parameter is not present in **trace-information-elements** (or **internal-trace-information-elements**) on probes and reports.

If the MD (or MTA) supplying the **trace-information-element** (or **internal-trace-information-element**) subjects a message to conversion, the **converted-encoded-information-types** resulting from the conversion is also included in the **trace-information-element** (or **internal-trace-information-element**). For a probe, an MD that would have converted the subject-message indicates the **encoded-information-types** the subject-message would contain after conversion in its **trace-information-element** (or **internal-trace-information-element**). This parameter is not present in **trace-information** (or **internal-trace-information-element**) on reports.

If the MD (or MTA) redirects a message or a probe (for any, but not necessarily all, of a message's or probe's recipients), **redirected** is indicated in the **trace-information-element** (or **internal-trace-information-element**). This parameter is not present in **trace-information** (or **internal-trace-information**) on reports.

If the MD (or MTA) expands a DL of a message or a probe, **dl-operation** is indicated in **trace-information-element** (or **internal-trace-information-element**). If the MD (or MTA) is a DL expasion-point and replaces its own **OR-name** in the **report-destination-name** of a report with another **OR-name** (see § 12.2.1.3.1.2), **dl-operation** is indicated in the **trace-information-element** (or **internal-trace-information-element**) of the report.

Loop detection and suppression is done by an MD (or MTA) when it receives a message, probe or report from another MD (or MTA). Messages, probes and reports may legitimately re-enter an MD (or MTA) for several reasons (**rerouted**, etc) and consequently a message, probe or report may have several disjoint **trace-information-elements** (or **internal-trace-information-elements**) from the same MD (or MTA). Each time a message, probe or report is transferred through an MD (or MTA) the generation of **trace-information-elements** (or **internal-trace-information-elements**) is performed as follows:

i) one **trace-information-element** (or **internal-trace-information-element**) is added, marked as **relayed**;

ii) if a rerouting attempt is to occur, then the **trace-information-element** (or **internal-trace-information-element**) added in i) is modified to **rerouted** (and the number of **trace-information-element** (or **internal-trace-information-elements**) added by the MD (or MTA) for this traversal of the MD (or MTA) remains at one);

iii) if subsequent attempts to reroute occur, then a new **trace-information-element** (or **internal-trace-information-element**) is added (marked as **rerouted**) to reflect each new rerouting attempt.

Several rerouting attempts to the same MD (or MTA) may occur.

Each **trace-information-element** (or **internal-trace-information-element**) added by an MD (or MTA) may contain indications of **additional-actions** performed by the MD (or MTA) on the message or probe (i.e., **deferred-time** (not present in **trace-information** (or **internal-trace-information**) on probes), **converted-encoded-information-types**, **redirected** or **dl-operation**).

13 Message transfer agent abstract syntax definition

The abstract-syntax of the MTA abstract service is defined in Figure 4/X.411.

The abstract-syntax of the MTA abstract service is defined using the abstract syntax notation (ASN.1) defined in Recommendation X.208, and the absract service definition conventions defined in Recommendation X.407.

The abstract-syntax definition of the MTA abstract service has the following major parts:

- *Prologue*: declaration of the exports from, and imports to, the MTA abstract service module (Figure 4/X.411, Part 1).
- *MTS refinement, objects and ports*: refinement of the MTS object, and definitions of the MTA object and the transfer-port (Figure 4/X.411, Part 2).
- *MTA-bind and MTA-unbind*: definitions of the MTA-bind and MTA-unbind used to establish and release associations between MTAs (Figure 4/X.411, Part 3).
- *Transfer ports*: definitions of the transfer-port abstract-operations: message-transfer, probe-transfer and report-transfer (Figure 4/X.411, Part 4).
- *Message transfer envelope*: definition of the message-transfer-envelope (Figure 4/X.411, Parts 5 and 6).
- *Probe transfer envelope*: definition of the probe-transfer-envelope (Figure 4/X.411, Part 7).
- *Report transfer envelope and content*: definitions of the report-transfer-envelope and report-transfer-content (Figure 4/X.411, Part 8).
- *Envelope and report content fields*: definitions of envelope and report content fields (Figure 4/X.411, Parts 9 and 10).
- *Extension fields*: definitions of extension-fields (Figure 4/X.411, Parts 11 and 12).
- *Common parameters types*: definitions of common parameter types (Figure 4/X.411, Part 13).

Note – The module implies a number of changes to the P1 protocol defined in Recommendation X.411 (1984). These changes are highlighted by means of underlining.

Each **extension-field** defined in Figure 4/X.411 (Parts 12 and 13) carries with it an indication of its **criticality** for submission, transfer and delivery. The criticality mechanism is described in § 9.1, and the procedures related to **extension-fields** and their **criticality** indications are further defined in § 14.

```
MTAAbstractService { joint-iso-ccitt mhs-motis(6) mts(3) modules(0) mta-abstract-service(2) }
DEFINITIONS IMPLICIT TAGS ::=
BEGIN


-- Prologue
-- Exports everything


IMPORTS
        -- Abstract service macros
        REFINE, OBJECT, PORT, ABSTRACT-BIND, ABSTRACT-UNBIND, ABSTRACT-OPERATION
                FROM AbstractServiceNotation { joint-iso-ccitt mhs-motis(6) asdc(2) modules(0)
                    notation(1) }

        -- MTS abstract service parameters
        mTS, submission, delivery, administration, InitiatorCredentials, SecurityContext,
        ResponderCredentials, OriginalEncodedInformationTypes, ContentTypes, ContentIdentifier,
        Priority, PerMessageIndicators, DeferredDeliveryTime, CountryName, AdministrationDomainName,
        PrivateDomainIdentifier, ExplicitConversion, ContentLength, ConvertedEncodedInformationTypes,
        ReportType, SupplementaryInformation, EXTENSION, EXTENSIONS, recipient-reassignment-prohibited,
        dl-expansion-prohibited, conversion-with-loss-prohibited, latest-delivery-time,
        requested-delivery-method, physical-forwarding-prohibited, physical-forwarding-address-request,
        physical-delivery-modes, registered-mail-type, recipient-number-for-advice, physical-rendition-attributes,
        originator-return-address, physical-delivery-report-request, originator-certificate, message-token,
        content-confidentiality-algorithm-identifier, content-integrity-check, message-origin-authentication-check,
        message-security-label, proof-of-delivery-request, content-correlator, probe-origin-authentication-check,
        redirection-history, dl-expansion-history, originator-and-dl-expansion-history, reporting-dl-name,
        physical-forwarding-address, recipient-certificate, proof-of-delivery, reporting-MTA-certificate,
        report-origin-authentication-check, Content, MTSIdentifier, GlobalDomainIdentifier, MTAName, Time,
        ORAddressAndOptionalDirectoryName
                FROM MTSAbstractService { joint-iso-ccitt mhs-motis(6) mts(3) modules(0)
                    mts-abstract-service(1) }

        -- Object identifiers
        id-ot-mta, id-pt-transfer
                FROM MTSObjectIdentifiers { joint-iso-ccitt mhs-motis(6) mts(3) modules(0)
                    object-identifiers(0) }

        -- Upper bounds
        ub-bit-options, ub-dl-expansions, ub-integer-options, ub-recipients, ub-redirections, ub-transfers
                FROM MTSUpperBounds { joint-iso-ccitt mhs-motis(6) mts(3) modules(0)
                    upper-bounds(3) };
```

FIGURE 4/X.411 (Part 1 of 13)

Abstract syntax definition of the MTA abstract service

```
-- MTS refinement

MTSRefinement ::= REFINE mTS AS
        mTA RECURRING
                submission      [S] VISIBLE
                delivery        [S] VISIBLE
                administration  [S] VISIBLE
                transfer            PAIRED WITH mTA


-- Objects

mTA OBJECT
        PORTS { submission [S], delivery [S], administration [S], transfer }
        ::= id-ot-mta


-- Ports

transfer PORT
        ABSTRACT OPERATIONS { MessageTransfer, ProbeTransfer, ReportTransfer }
        ::= id-pt-transfer
```

FIGURE 4/X.411 (Part 2 of 13)

Abstract syntax definition of the MTA abstract service

```
-- MTA-bind and MTA-unbind

MTABind ::= ABSTRACT-BIND
        TO { transfer }
        BIND
        ARGUMENT CHOICE {
                NULL,           -- if no authentication is required
                [1] SET {       -- if authentication is required
                    initiator-name [0] MTAName,
                    initiator-credentials [1] InitiatorCredentials,
                    security-context [2] SecurityContext OPTIONAL } }
        RESULT CHOICE {
                NULL,           -- if no authentication is required
                [1] SET {       -- if authentication is required
                    responder-name [0] MTAName,
                    responder-credentials [1] ResponderCredentials } }
        BIND-ERROR INTEGER {
                busy (0),
                authentication-error (2),
                unacceptable-dialogue-mode (3)
                unacceptable-security-context (4) } (0 .. ub-integer-options)

MTAUnbind ::= ABSTRACT-UNBIND
        FROM { transfer }
```

FIGURE 4/X.411 (Part 3 of 13)

Abstract syntax definition of the MTA abstract service

```
-- Transfer port

MessageTransfer ::= ABSTRACT-OPERATION
        ARGUMENT Message

ProbeTransfer ::= ABSTRACT-OPERATION
        ARGUMENT Probe

ReportTransfer ::= ABSTRACT-OPERATION
        ARGUMENT Report

Message ::= SEQUENCE {
        envelope MessageTransferEnvelope,
        content Content }

Probe ::= ProbeTransferEnvelope

Report ::= SEQUENCE {
        envelope ReportTransferEnvelope,
        content ReportTransferContent }
```

FIGURE 4/X.411 (Part 4 of 13)

Abstract syntax definition of the MTA abstract service

```
-- Message transfer envelope

MessageTransferEnvelope ::= SET {
        COMPONENTS OF PerMessageTransferFields,
        per-recipient-fields [2] SEQUENCE SIZE (1..ub-recipients) OF
                PerRecipientMessageTransferFields }

PerMessageTransferFields ::= SET {
        message-identifier MessageIdentifier,
        originator-name OriginatorName,
        original-encoded-information-types OriginalEncodedInformationTypes OPTIONAL,
        content-type ContentType,
        content-identifier ContentIdentifier OPTIONAL,
        priority Priority DEFAULT normal,
        per-message-indicators PerMessageIndicators DEFAULT { },
        deferred-delivery-time [0] DeferredDeliveryTime OPTIONAL,
        per-domain-bilateral-information [1] SEQUENCE OF PerDomainBilateralInformation OPTIONAL,
        trace-information TraceInformation,
        extension [3] EXTENSIONS CHOSEN FROM {
                recipient-reassignment-prohibited,
                dl-expansion-prohibited,
                conversion-with-loss-prohibited,
                latest-delivery-time,
                originator-return-address,
                originator-certificate,
                content-confidentiality-algorithm-identifier,
                message-origin-authentication-check,
                message-security-label,
                content-correlator,
                dl-expansion-history,
                internal-trace-information} DEFAULT {} }
```

FIGURE 4/X.411 (Part 5 of 13)

Abstract syntax definition of the MTA abstract service

```
PerRecipientMessageTransferFields ::= SET {
        recipient-name RecipientName,
        originally-specified-recipient-number [0] OriginallySpecifiedRecipientNumber,
        per-recipient-indicators [1] PerRecipientIndicators,
        explicit-conversion [2] ExplicitConversion OPTIONAL,
        extension [3] EXTENSIONS CHOSEN FROM {
                originator-requested-alternate-recipient,
                requested-delivery-method,
                physical-forwarding-prohibited,
                physical-forwarding-address-request,
                physical-delivery-modes,
                registered-mail-type,
                recipient-number-for-advice,
                physical-rendition-attributes,
                physical-delivery-report-request,
                message-token,
                content-integrity-check,
                proof-of-delivery-request,
                redirection-history } DEFAULT {} }
```

FIGURE 4/X.411 (Part 6 of 13)

Abstract syntax definition of the MTA abstract service

```
-- Probe transfer envelope

ProbeTransferEnvelope ::= SET {
        COMPONENT OF PerProbeTransferFields,
        per-recipient-field [2] SEQUENCE SIZE (1 .. ub-recipient) OF PerRecipientProbeTransferFields }

PerProbeTransferFields ::= SET {
        probe-identifier ProbeIdentifier,
        originator-name OriginatorName,
        original-encoded-information-types OriginalEncodedInformationTypes OPTIONAL,
        content-type-ContentType,
        content-identifier ContentIdentifier OPTIONAL,
        content-length [0] ContentLength OPTIONAL,
        per-message-indicators PerMessageIndicators DEFAULT {},
        per-domain-bilateral-information [1] SEQUENCE SIZE (1 .. ub-transfers) OF
                PerDomainBilateralInformation OPTIONAL,
        trace-information TraceInformation,
        extensions [3] EXTENSIONS CHOSEN FROM {
                recipient-reassignment-prohibited,
                dl-expansion-prohibited,
                conversion-with-loss-prohibited,
                originator-certificate,
                message-security-label,
                content-correlator,
                probe-origin-authentication-check,
                dl-expansion-history,
                internal-trace-information} DEFAULT {} }

PerRecipientProbeTransferFields ::= SET {
        recipient-name RecipientName,
        originally-specified-recipient-number [0] OriginallySpecifiedRecipientNumber,
        per-recipient-indicators [1] PerRecipientIndicators,
        explicit-conversion [2] ExplicitConversion OPTIONAL,
        extensions [3] EXTENSIONS CHOSEN FROM {
                originator-requested-alternate-recipient,
                requested-delivery-method,
                physical-rendition-attributes,
                redirection-history } DEFAULT {} }
```

FIGURE 4/X.411 (Part 7 of 13)

Abstract syntax definition of the MTA abstract service

```
-- Report transfer envelope

ReportTransferEnvelope ::= SET {
        report-identifier ReportIdentifier,
        report-destination-name ReportDestinationName,
        trace-information TraceInformation,
        extensions [1] EXTENSIONS CHOSEN FROM {
                message-security-label,
                originator-and-DL-expansion-history,
                reporting-DL-name,
                reporting-MTA-certificate,
                report-origin-authentication-check,
                internal-trace-information} DEFAULT {} }


-- Report transfer content

ReportTransferContent ::= SET {
        COMPONENT OF PerReportTransferFields,
        per-recipient-fields [0] SEQUENCE SIZE (1 . . ub-recipients) OF
                PerRecipientReportTransferFields }

PerReportTransferFields ::= SET {
        subject-identifier SubjectIdentifier,
        subject-intermediate-trace-information SubjectIntermediateTraceInformation OPTIONAL,
        original-encoded-information-types OriginalEncodedInformationTypes OPTIONAL,
        content-type ContentType OPTIONAL,
        content-identifier ContentIdentifier OPTIONAL,
        returned-content [1] Content OPTIONAL,
        additional-information [2] AdditionalInformation OPTIONAL,
        extensions [3] EXTENSIONS CHOSEN FROM {
                content-correlator } DEFAULT {} }

PerRecipientReportTransferFields ::= SET {
        actual-recipient-name [0] ActualRecipientName,
        originally-specified-recipient-number [1] OriginallySpecifiedRecipientNumber,
        per-recipient-indicator [2] PerRecipientIndicators,
        last-trace-information [3] LastTraceInformation,
        originally-intended-recipient-name [4] OriginallyIntendedRecipientName OPTIONAL,
        supplementary-information [5] SupplementaryInformation OPTIONAL,
        extensions [6] EXTENSIONS CHOSEN FROM {
                redirection-history,
                physical-forwarding-address,
                recipient-certificate,
                proof-of-delivery } DEFAULT {} }
```

FIGURE 4/X.411 (Part 8 of 13)

Abstract syntax definition of the MTA abstract service

```
-- Envelope and report content fields
MessageIdentifier ::= MTSIdentifier
OriginatorName ::= ORAddressAndOptionalDirectoryName
PerDomainBilateralInformation ::= SEQUENCE {
        country-name CountryName,
        CHOICE {
                administration-domain-name AdministrationDomainName,
                SEQUENCE {
                    administration-domain-name [0] AdministrationDomainName,
                    private-domain-identifier [1] PrivateDomainIdentifier OPTIONAL } },
        bilateral-information BilateralInformation }
BilateralInformation ::= ANY        --maximum ub-bilateral-info octets including all encoding
RecipientName ::= ORAddressAndOptionalDirectoryName
OriginallySpecifiedRecipientNumber ::= INTEGER (SIZE (1 . . ub-recipients))
PerRecipientIndicators ::= BIT STRING {
        responsibility (0),
        -- reponsible 'one', not-responsible 'zero'
        originating-MTA-report (1),
        originating-MTA-non-delivery-report (2),
        -- either originating-MTA-report, or originating-MTA-non-delivery-report, or both, shall be 'one':
        -- originating-MTA-report bit 'one' requests a 'report';
        -- originating-MTA-non-delivery-report bit 'one' requests a 'non-delivery-report';
        -- both bits 'one' requests and 'audited-report';
        -- bits 0-2 'don't care' for Report Transfer Content
        originator-report (3),
        originator-non-delivery-report (4),
        -- at most one bit shall be 'one':
        -- originator-report bit 'one' requests a 'report';
        -- originator-non-delivery-report bit 'one' requests a 'non-delivery-report';
        -- both bits 'zero' requests 'no-report'
        reserved-5 (5),
        reserved-6 (6),
        reserved-7 (7),
        -- reserved-bits 5-7 shall be 'zero' -- } (SIZE (8 . . ub-bit-options))
ProbeIdentifier ::= MTSIdentifier
```

FIGURE 4/X.411 (Part 9 fo 13)

Abstract syntax definition of the MTA abstract service

```
ReportIdentifier ::= MTSIdentifier
ReportDestinationName ::= ORAddressAndOptionalDirectoryName
SubjectIdentifier ::= MessageOrProbeIdentifier
MessageOrProbeIdentifier ::= MTSIdentifier
SubjectIntermediateTraceInformation ::= TraceInformation
AdditionalInformation ::= ANY        -- maximum ub-additional-info octets including all enconding
ActualRecipientName ::= ORAddressAndOptionalDirectoryName
LastTraceInformation ::= SET {
        arrival-time [0] ArrivalTime,
        converted-encoded-information-type ConvertedEncodedInformationTypes OPTIONAL,
        report-type [1] ReporType }
OriginallyIntendedRecipientName ::= ORAddressAndOptionalDirectoryName
```

FIGURE 4/X.411 (Part 10 of 13)

Abstract syntax definition of the MTA abstract service

```
-- Extension fields

originator-requested-alternate-recipient EXTENSION
        OriginatorRequestedAlternateRecipient
        ::= 2

OriginatorRequestedAlternateRecipient ::= ORAddressAndOptionalDirectoryName

internal-trace-information EXTENSION
        InternalTraceInformation
        ::= 38
```

FIGURE 4/X.411 (Part 11 of 13)

Abstract syntax definition of the MTA abstract service

```
InternalTraceInformation ::= SEQUENCE SIZE (1 . . ub-transfers) OF InternalTraceInformationElement

InternalTraceInformationElement ::= SEQUENCE {
        global-domain-identifier GlobalDomainIdentifier,
        mta-name MTAName,
        mta-supplied-information MTASuppliedInformation }

MTASuppliedInformation ::= SET {
        arrival-time [0] ArrivalTime,
        routing-action [2] RoutingAction,
        attempted CHOICE {
                mta MTAName,
                domain GlobalDomainIdentifier } OPTIONAL,
        -- additional-actions-- COMPONENTS OF InternalAdditionalActions }

InternalAdditionalActions ::= AdditionalActions
```

FIGURE 4/X.411 (Part 12 of 13)

Abstract syntax definition of the MTA abstract service

```
-- Common parameter types

TraceInformation ::= [APPLICATION 9] SEQUENCE (SIZE (1 . . ub-transfers) OF TraceInformationElement

TraceInformationElement ::= SEQUENCE {
        global-domain-identifier GlobalDomainIdentifier,
        domain-supplied-information DomainSuppliedInformation }

DomainSuppliedInformation ::= SET {
        arrival-time [0] ArrivalTime,
        routing-action [2] RoutingAction,
        attempted-domain GlobalDomainIdentifier OPTIONAL,
        -- additional-actions-- COMPONENT OF AdditionalActions }

AdditionalActions ::= SET {
        deferred-time [1] DeferredTime OPTIONAL,
        converted-encoded-information-types ConvertedEncodedInformationTypes OPTIONAL,
        other-actions [3] OtherActions DEFAULT {} }

RoutingAction ::= ENUMERATED {
        relayed (0),
        rerouted (1) }

DeferredTime ::= Time

ArrivalTime ::= Time

OtherActions ::= BIT STRING {
        redirected (0),
        dl-operation (1) } (SIZE (0 . . ub-bit-options))

END  -- of MTA abstract service
```

FIGURE 4/X.411 (Part 13 of 13)

Abstract syntax definition of the MTA abstract service

14 Procedures for distributed operation of the MTS

This paragraph specifies the procedures for distributed operation of the MTS, which are performed by MTAs. Each MTA individually performs the procedures described below; the collective action of all MTAs provides the MTS Abstract Service to the users of the MTS.

Although the procedures include most of the important actions required of an MTA, considerable detail has been omitted for clarity of exposition and to avoid unnecessary redundancy. The abstract-service definitions should be consulted for a definitive treatment of MTA actions.

14.1 *Overview of the MTA model*

14.1.1 *Organization and modelling technique*

The description of procedures for a single MTA is based on the model shown in Figures 5/X.411 through 11/X.411 and described below. It should be noted that the model is included for expositional purposes only and is not intended to constrain in any way the implementation of an MTA.

Neither the procedures shown nor the order of processing steps in them necessarily imply specific characteristics of an actual MTA.

The model distinguishes between *modules* and *procedures*. *Modules*, in the sense used here, are autonomous processing entities which can be invoked by other modules or by events external to the MTA, and which can in turn invoke other modules or generate external events. Modules are not bound together by an explicitly described control structure; rather the control structure among modules arises from the pattern of cross invocations. Modules correspond to *objects* in the sense of object-oriented programming.

Procedures are used here in the conventional programming sense. Procedures are task or function oriented. Procedures can call other procedures, subroutine fashion, with control returning to the calling procedure when the called procedure has completed. Such calls can be nested to arbitrary depth, and a procedure can call itself recursively. Procedures are bound together by explicitly defined control structures built from procedure calls and such conventional programming devices as iteration and conditional execution.

In the model procedures exist within modules. Each module contains at least one procedure and can contain several. In the latter case, the procedures and governing control structure are described explicitly. In the former case the existence of a module's single procedure is usually treated as implicit.

Using these modelling techniques, an MTA application process can be refined as follows: for each abstract-operation (whether consumer or supplier) that can exist between an MTA and the MTS-users it serves, or between an MTA and the other MTAs with which it cooperates there is a single module called an *external module*. The set of external module is responsible for the input and output of messages, probes, and reports into and out of the MTA and for the support of such operations as MTS-bind, MTS-unbind, Register, Submission-control and Delivery-control. The external modules are shown in Figure 5/X.411 and described in §§ 14.5 through 14.10, grouped by port.

In order to perform the various abstract-operations for which it is responsible, an MTA must perform certain processing operations on each message, probe, or report that enters, or originates within it. In the model these are the province of *internal modules*, shown in Figure 6/X.411 and described in §§ 14.2 through 14.4.

The external and internal modules relate to one another as follows: an external module comunicates only with an internal module, and not with another external module or directly with a procedure within an internal module. Thus, the internal modules not only support the bulk of processing within an MTA, but also serve as links between its external modules. In addition to the internal modules Figure 6/X.411 also shows the external modules with which they communicate.

The MTA is event driven in that it remains quiescent until an event is detected on one of its ports. Many events, such as the invocation of a MTS-bind, Submission-control, Delivery-control or Register abstract-operation by an MTS-user or another MTA, are dealt with directly and completely by the module assigned to that

abstract-operation. However other events trigger processing that can reverberate through the MTA, endure over time and ultimately trigger one or more output events. Is is these events that engage the internal processing modules. They are:

a) a message or probe originated by a locally supported MTS-user enters via the submission-port;

b) a message, probe or report relayed from another MTA enters via the transfer-port.

Because the processing within an MTA can become rather complex, especially for messages with multiple recipients, the model assumes, as an internal bookkeeping device, that each message carries with it a set of instructions, one for the message as a whole, and one for each recipient. These instructions help guide a message through the processing steps and convey information between the modules and procedures internal to the MTA.

Note 1 – The procedures described herein focus on the processing of a single message. This is adequate in all but one respect: the queuing of messages and the relative prioriry of procedure invocation are driven explicitly by the argument **priority** in case of a message which enters via the submission – or the transfer-port, or implicitly (of urgent priority) in the case of a report or a probe which is generated internally or enters via the transfer-port.

Note 2 – An MTA can specify several default delivery time windows for each message priority e.g. those values defined in the F.400 series Recommendations. The MTS and therefore each MTA involved should take such values into account during message processing. For example, the MTA can apply a maximum delivery deadline. If that time period expires prior to delivery, the MTA generates a non-delivery-report and discards the message. The required actions in this case are identical to the actions required when **latest-delivery-time** is reached.

Note 3 – The discussion of trace-information is incomplete due to its complex nature. Some important details are highlighted but the complete and definitive treatment of trace-information appears in § 12.3.1.

14.2 *Deferred delivery module*

This module provides the Deferred Delivery element-of-service. It is invoked by the Message-submission and Message-in modules which pass a message to be checked for deferred delivery request and held if necessary. It invokes the Main module, passing on the message upon completion of its single internal procedure.

14.2.1 *Deferred delivery procedure*

14.2.1.1 *Arguments*

A message to be checked for deferred delivery request and held if necessary.

14.2.1.2 *Results*

The message is returned after expiration of the **deferred-delivery-time**. If deferred occurred, an arrival timestamp accompanies the message.

14.2.1.3 *Errors*

None.

14.2.1.4 *Procedure description*

The message is checked for presence of the **deferred-delivery-time** field. If absent the procedure returns the message and terminates. If present the **deferred-delivery-time** is checked against current time. If the **deferred-delivery-time** has expired, the procedure returns the message and terminates.

Otherwise, in the case of a relayed message, the MTA checks for a bilateral agreement obligating it to provide deferred delivery for this message. If absent the procedure returns the message and terminates.

Otherwise depending on bilateral agreement or intra-domain policy the current time is noted as the message arrival time and the message is held until expiration of the **deferred-delivery-time**. The message and timestamp are then returned as result. The procedure then terminates.

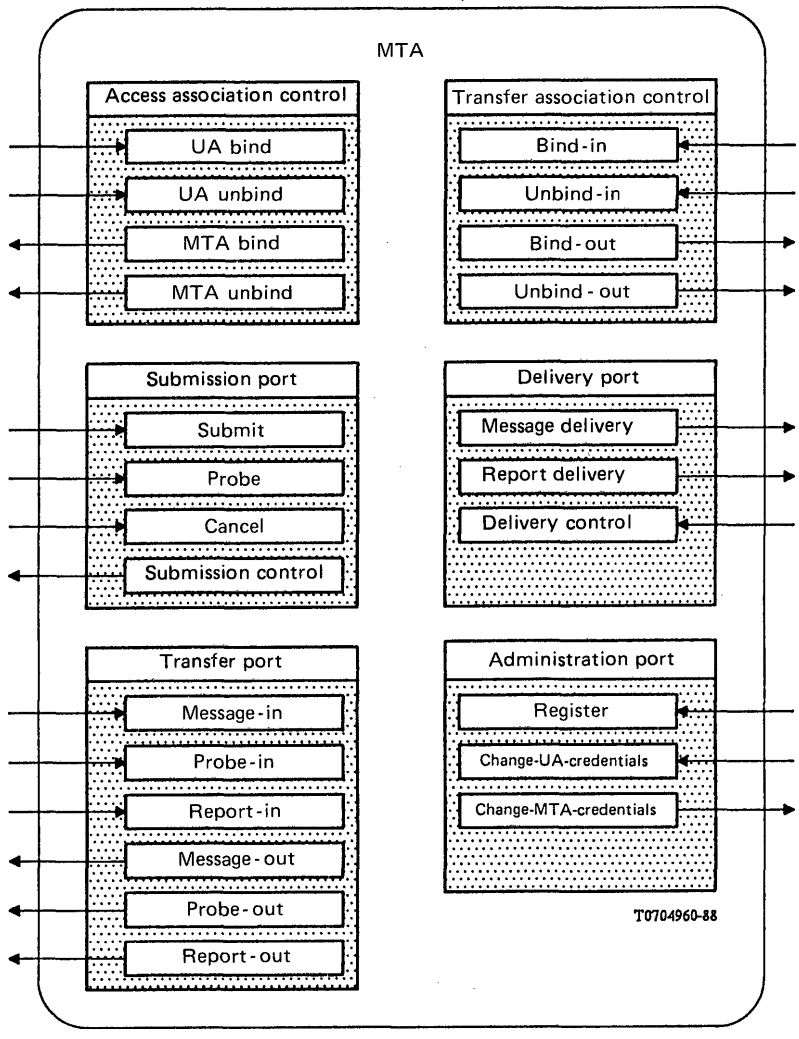


FIGURE 5/X.411

Ports and modules of an MTA

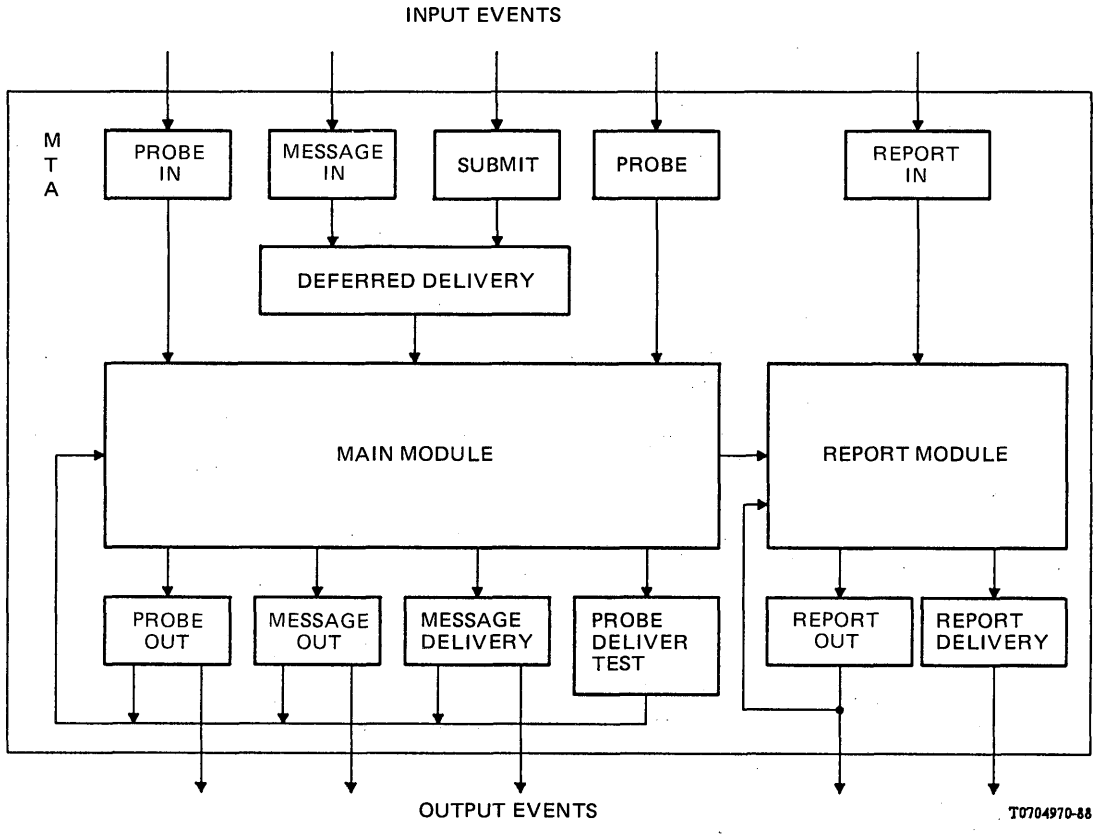


FIGURE 6/X.411

Relationship of internal and external modules

14.3 *Main module*

The Main module performs the bulk of processing on messages and probes entering the MTA. Figure 6/X.411 shows the relationships between the main module and the modules which it can invoke or be invoked by. The main module is subject to invocation by:

1) the Probe-in module, which passes a probe;
2) the Deferred-delivery module, which passes a message;
3) the Probe module, which passes a probe.

In the case of an error condition or the need for a positive delivery report, the main module can also be invoked by:

4) the Message-out module, which passes a message with per-message instruction indicating the problem encountered;
5) the Probe-out module, which passes a probe with per-message instruction indicating the problem encountered;
6) the Message-delivery module, which passes a message with per-recipient instruction indicating the problem(s) and/or success(es) encountered;
7) the Probe-delivery-test module, which passes a probe with per-recipient instructions indicating the problem(s) or success(es) encountered.

The Main module contains procedures which collectively, support the following functions:

- Trace processing
- Loop detection
- Routing and rerouting
- Recipient redirection
- Content conversion
- Distribution list expansion
- Message replication
- Origin authentication of messages and probes
- Name resolution.

The procedures that perform these functions are called by a single Control procedure that guides the processing of each message or probe received by the Main module. Figure 7/X.411 shows the organization of the Control and subsidiary procedures within the main module; Figure 8/X.411 shows the flow of information through these procedures.

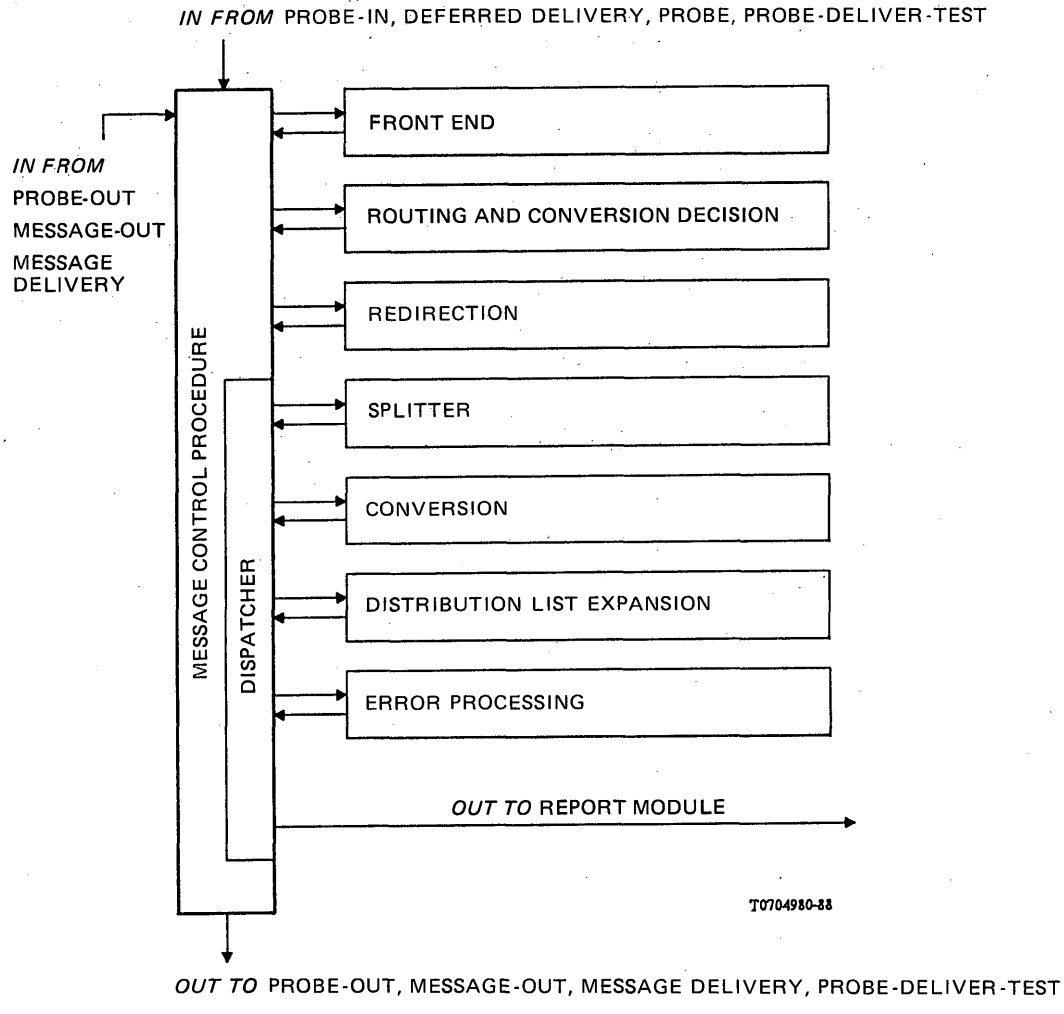
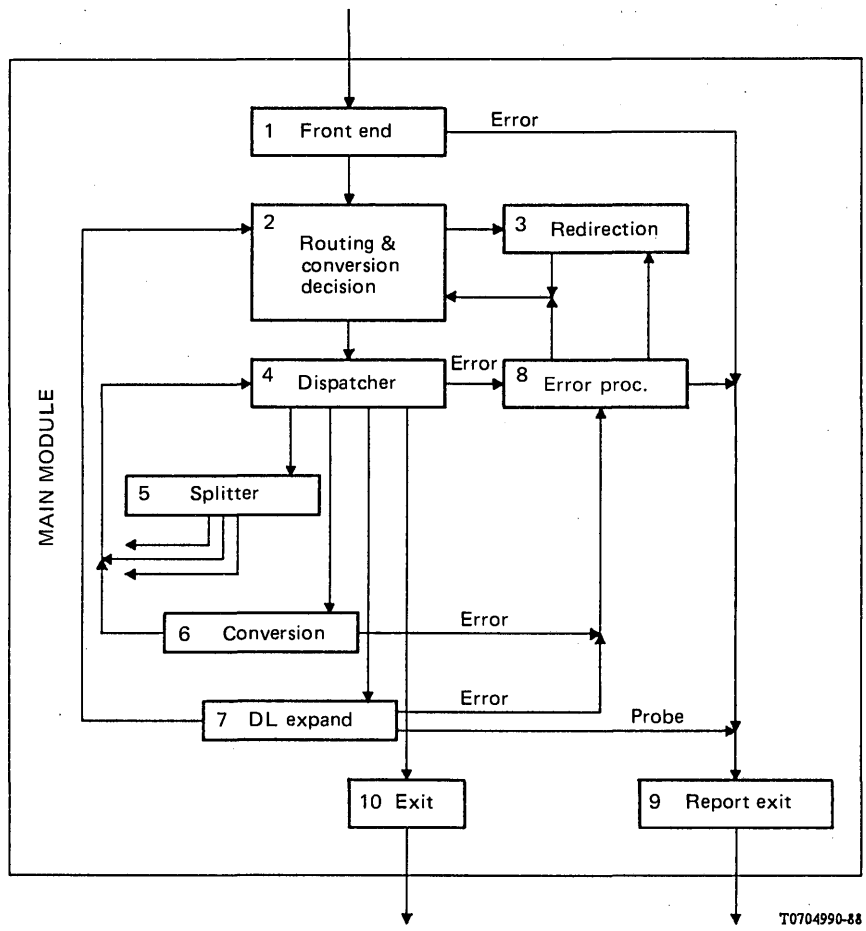


FIGURE 7/X.411

Organization of procedures within the main module



Note – Numbers in this Figure refer to the numbered steps in the control procedure's logic (§ 14.3.1.4).

FIGURE 8/X.411

Information flow within the main module

For each message or probe received, the Main module calls the Control procedure with that message or probe as argument. As result, the Control procedure returns one or more replicas of the message or probe with appropriate instructions attached. Depending on the nature of these instructions the Main module then invokes:

1) the message-out module, to which it passes each message with a per-message transfer instruction;
2) the probe-out module, to which it passes each probe with a per-message transfer instruction;
3) the message-delivery module, to which it passes each message with one or more per-recipient delivery instructions;
4) the probe-delivery-test module, to which it passes each probe with one or more per-recipient delivery instructions;
5) the report module, to which it passes each message or probe with a per-message instructions and/or one or more per-recipient instructions indicating report generation.

14.3.1 *Control procedure*

This procedure directs each incoming message or probe through the remaining procedures of the Main module. The overall flow of information is shown in Figure 8/X.411.

14.3.1.1 *Arguments*

One of the following (these arguments correspond to the messages and probes that can be passed to the Main module upon invocation):

1) a message or probe without instructions (from the probe-in or probe module);
2) a message without instructions but with optional arrival timestamp (from the deferred-delivery module);
3) a message or probe with per-message instruction describing a transfer problem (from the message-out or probe-out module);
4) a message or probe with per-recipient instructions describing delivery problems or successes (from the message-delivery or probe-delivery-test module).

14.3.1.2 *Results*

1) One or more replicas of the message or probe argument each accompanied by a per-message instruction indicating transfer, and/or
2) one or more replicas of the message or probe argument each accompanied by one or more per-recipient instructions indicating delivery or delivery test, and/or
3) one or more replicas of the message or probe argument each accompanied by one or more per-recipient instructions indicating report generation.

14.3.1.3 *Errors*

None. Error conditions are accounted for in the results described above.

14.3.1.4 *Procedure description*

1) A message or probe without instructions:

   The Front-end procedure is first called to perform trace initialization and several per message checks such as message expiration and routing loop detection.

   Upon a return with report instruction indicating a problem with the message processing continues at step 9.

   On all other returns processing continues below.
2) Routing-and-conversion-decision procedure is called to compute per-recipient routing and conversion instructions. (These are complete instructions that will direct the message or probe through the remainder of the procedures.)

   If a redirection instruction is indicated (e.g., recipient-requested-alternate-recipient), processing continues at step 3.

   Otherwise, processing continues at step 4 (dispatcher).
3) Redirection is called. Upon successful return, processing continues at step 2.

   In the case of an unsuccessful return, processing continues at step 8 (error-handler).
4) Dispatcher. The dispatcher acts on the generated instructions and passes control to the first of the following procedures that is applicable:
   - splitting (step 5);
   - conversion (step 6);
   - distribution-list-expansion (step 7);
   - error-processing (step 8) in case the decision process encountered a problem, e.g., routing error;
   - exit (step 10).
5) Splitter is called for replication as required by the per-recipient instructions generated in routing-and-conversion-decision procedure. For each replica processing continues individually at step 4 (dispatcher).
6) Conversion is called for each message or probe needing conversion.

   Upon successful return of the message or probe, processing continues at step 4 (dispatcher).

   Upon return with report instruction indicating a conversion error, processing continues at step 8 (error-handler).

7) The DL-expansion procedure is called.

   Upon successful return of a message, processing continues at step 2 so that the recipients resulting from DL expansion can be properly dealt with.

   If a copy of the message with delivery report instructions is returned, in place of or in addition to the above return, its processing continues at step 9.

   A probe returning successfully will have report instructions; processing continues at step 9 (report-generation).

   Upon return of a message or probe with report instruction indicating DL expansion error-processing continues at step 8.

8) This is the collection point that processing reaches upon detection that a message or probe cannot be handled by the main line procedures. The error-processing procedure is called to seek another delivery method or an alternate-recipient. Upon successful return the error-processing procedure indicates the new recipient in an instruction to the Routing-and-conversion-decision procedure (step 2), where processing continues.

   If redirection is possible, the message or probe is passed to the report generator (step 9).

9) The control procedure terminates at this point and returns a message or probe with report generation instructions.

10) When a message or probe reaches this point the control procedure terminates.

14.3.2 *Front-end procedure*

This procedure performs trace initialization, detection of message expiration, initial security check, loop detection, and criticality check.

14.3.2.1 *Arguments*

A message or probe and an optional arrival timestamp.

14.3.2.2 *Results*

The message, or probe with initialized trace information for this MTA.

14.3.2.3 *Errors*

The message or probe with report generation instructions detailing the problem encountered.

14.3.2.4 *Procedure description*

1) If the message has crossed a domain boundary, a **trace-information-element** for this domain is added with **relay** as action. If an arrival time accompanies the message, then delivery deferral has occurred and **deferred-time** is set to the current time and **arrival-time** is set to the accompanying timestamp value. Otherwirse no deferral has occurred and the **arrival-time** is set to the current time. An **internal-trace-information-element** is also added whether or not the message has crossed a domain boundary.

2) If required by the security policy in force and/or if the **message-origin-authentication-check** is incorrect, the procedure returns a report generation instruction. The values of the **non-delivery-reason-code** and **non-delivery-diagnostic-code** are set to **unable-to-transfer**, and **secure-messaging-error**, respectively.

3) If any of the extension fields is marked critical for relaying but is not semantically understood by the MTA, the procedure returns a report generation instruction. The **non-delivery-reason-code** is set to **transfer-failure** and the **non-delivery-diagnostic-code** to **unsupported-critical-function**. The procedure then terminates.

4) If the **latest-delivery-time** has passed, or the system's maximum transit time has elapsed for the message's **priority**, the procedure returns a report generation instruction. The **non-delivery-reason-code** is set to **unable-to-transfer** and the **non-delivery-diagnostic-code** is set to **maximum-time-expired**. The procedure then terminates.

5) Loop detection is performed. The loop detection algorithm is beyond the scope of this Recommendation. However, an example of a combined routing and loop detection algorithm is given in § 14.3.11. If a loop is detected, the procedure returns a report generation instruction. The **non-delivery-reason-code** is set to **transfer-failure** and the **non-delivery-diagnostic-code** is set to **loop-detected**. The procedure then terminates.

14.3.3 *Routing-and-conversion-decision procedure*

For each of a message or probe's recipients for which the MTA is responsible, this procedure determines the routing and conversion actions, if any, to be taken by this MTA. The actions are recorded as per-recipient instructions associated with the message. The actions are subsequently carried out by other sub-procedures within the internal procedure, or elsewhere in the MTA.

Note – This procedure may be called multiple times for any particular message. In such cases, the procedure ignores per recipient instructions generated by previous calls to this procedure which have not yet been acted upon elsewhere.

14.3.3.1 *Arguments*

1) A message or probe with **responsibility** true for those recipients of concern to this MTA.

14.3.3.2 *Results*

The message or probe that formed the procedure's argument plus new or revised per-recipient instructions indicating what routing and possible conversion action should be taken by this MTA.

14.3.3.3 *Errors*

None. Error conditions, if any, are noted in the per-recipient instructions.

14.3.3.4 *Procedure description*

Each recipient is considered in turn. If **responsibility** is false, the recipient is ignored. Otherwise, the Routing-decision and Conversion-decision procedures are called in turn for this recipient. When all recipients have been considered in this way the procedure terminates. See Figure 9/X.411.

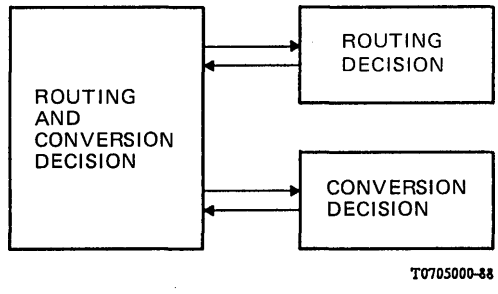


FIGURE 9/X.411

Organization of procedures within routing and conversion decision procedure

14.3.4 *Routing-decision procedure*

This procedure generates a routing instruction for a single message recipient.

14.3.4.1 *Arguments*

1) A message recipient plus the per-recipient instruction, if any, applicable to this recipient.

2) The per-message instruction, if any, applicable to this message. Other message fields are also accessible to the procedure as required.

14.3.4.2 *Results*

A new or possibly revised routing instruction applicable to this recipient. Possible instructions are:

a) relay to another MTA;

b) deliver to a local recipient;

c) expand the distribution list represented by this recipient;

d) generate a report indicating delivery failure. The **non-delivery-reason-code** and **non-delivery-diagnostic-code** are included in the instruction;

e) redirect to a recipient specified alternate recipient.

14.3.4.3 *Errors*

None. Error conditions are recorded in the routing instruction.

14.3.4.4 *Procedure description*

The procedure is described in the following steps.

Note – To ensure the security-policy is not violated during routing, the **message-security-label** should be checked as appropriate against the **security-context**.

1) If there is a per-message instruction indicating a previous relay failure, then the procedure attempts to compute an alternate next hop destination for this recipient. The choice of routing algorithm is beyond the scope of this Recommendation. However, an example of an applicable algorithm is contained in clause 14.3.11. If successful, then the message's **internal-trace-information** is updated with a **rerouted** routing-action to reflect the fact that the message has been re-routed (see § 12.3.1). If the message was to have crossed a domain boundary then the **trace-information** is also updated accordingly. The procedure returns a relay instruction to the alternate destination and terminates.

   If no alternate next hop is available or all available next hops have already been tried unsuccessfully or prohibited, then the procedure returns a report generation instruction for this recipient. The **non-delivery-reason-code** is set to **transfer-failure** and the **non-delivery-diagnostic-code** is set as appropriate to the realy failure encountered. The procedure then terminates.

2) If the per recipient instruction indicates a delivery failure, then the procedure returns a report generation instruction for this recipient. The **non-delivery-reason-code** and **non-delivery-diagnostic-code** are those supplied by the Message-delivery or Report-delivery procedure. The procedure then terminates.

3) If the recipient is a distribution list for which this MTA serves as expansion point, then the message's **DL-expansion-prohibited** argument is examined. If the value is **DL-expansion-allowed** then the procedure returns a routing instruction (subject to the security-policy in force) to expand the distribution list and terminates.

   If the value is **DL-expansion-prohibited**, or the security prohibits the use of a DL, then the procedure returns a report generation instruction for this recipient. The **non-delivery-reason-code** is set to **unable-to-transfer** and **non-delivery-diagnostic-code** to **DL-expansion-prohibited**. The procedure then terminates.

   In all cases other than the above, the following steps are taken.

4) If the recipient appears to be local, that is, an MTS-user directly supported by this MTA, then the following steps are taken.

   a) The **OR-address** is checked to ensure that it unambiguously specifies an actual local recipient. Otherwise the procedure returns a report generation instruction for this recipient. The **non-delivery-reason-code** is set to **unable-to-transfer** and the **non-delivery-diagnostic-code** is set to **unrecognized-OR-name** or **ambiguous-OR-name** as appropriate. The procedure then terminates.

   b) If the **OR-address** unambiguously specifies an actual local recipient, then the recipient registration parameters are checked for recipient-requested-alternate-recipient. In the determination of an alternate-recipient the **user-security-label** should be checked against the **message-security-label** to ensure no violation of the security-policy occurs.

If **recipient-assigned-alternate-recipient** is in effect, allowed by the **recipient-reassignment-prohibited** field, and permitted by the security-policy, then a redirection instruction is generated and the procedure terminates.

Otherwise the procedure returns a report instruction for this recipient and terminates. The **non-delivery-reason-code** is set to **unable-to-transfer** and the **non-delivery-diagnostic-code** is set as appropriate.

c) If **recipient-requested-alternate-recipient** is not in effect, then the message is checked against the recipient's remaining registration parameters. For example the message's content length is compared to the recipient's **deliverable-maximum-content-length**, the message's **content-type** to the recipient's **deliverable-content-types**, etc. If no problem is encountered, then the Routing-decision procedure returns a delivery instruction for this recipient and terminates.

If there is a problem between message and registration parameters, then the procedure returns a report generation instruction for this recipient. The **non-delivery-reason-code** is set to **unable-to-transfer** and the **non-delivery-diagnostic-code** is set as appropriate to the message problem encountered. The procedure then terminates.

5) If the recipient is not local to this MTA then the Routing-decision procedure attempts to determine a next hop instruction (subject to the security-policy in force) for this recipient. If successful, then a relay instruction to the next hop is returned and the procedure terminates.

If a next hop cannot be determined, then the procedure returns a report generation instruction for this recipient. The **non-delivery-reason-code** is set to **unable-to-transfer** and the **non-delivery-diagnostic-code** is set as appropriate to the problem encountered. The procedure then terminates.

14.3.5 *Conversion-decision procedure*

This procedure generates a conversion instruction for a single message recipient.

14.3.5.1 *Arguments*

1) A message or probe recipient plus the per-recipient instruction, if any, applicable to this recipient.
2) Other message fields are also considered by the procedure:
   a) **original-encoded-information-types**,
   b) **implicit-conversion-prohibited**,
   c) **conversion-with-loss-prohibited**,
   d) **explicit-conversion**.

14.3.5.2 *Results*

1) A content conversion instruction applicable to this recipient, and possibly,
2) a revised routing instruction indicating Relay-out or Probe-out to an MTA able to perform the required conversion, or, in lieu of 1 and 2 above,
3) an instruction to generate a report indicating delivery failure. The **non-delivery-reason-code** and **non-delivery-diagnostic-code** are included in the instruction.

14.3.5.3 *Errors*

None. Error conditions are recorded in the routing instruction.

14.3.5.4 *Procedure description*

Note – As the circumstances under which a particular MTA stages conversion are left for further study, it is impractical to describe a procedure to decide what EITs are required for conversion output. For example, if an intermediate MTA stages the conversion, there is no standardized way to know the EITs that the MTS-user can handle. Consequently the following clauses assume that the EITs for conversion are known to the MTA.

1) If explicit conversion is required for this recipient, the procedure starts at step 6.
2) If implicit conversion is required but the recipient has not subscribed to the implicit conversion facility, the procedure returns a negative report instruction with the **non-delivery-reason-code conversion-not-performed** and the non-delivery-diagnostic-code implicit-conversion-not-subscribed. The procedure then terminates.

3) If the required conversion is impractical, the procedure generates a negative report instruction with the **non-delivery-reason-code conversion-not-performed** and the **non-delivery-diagnostic-code conversion-impractical**. The procedure then terminates.

4) If conversion would be required but is prohibited for the message, the procedure generates a negative report instruction with the **non-delivery-reason-code conversion-not-performed** and the **non-delivery-diagnostic-code conversion-prohibited**. The procedure then terminates.

5) If the required conversion would cause a loss of information and the **conversion-with-loss-prohibited** field has the value **with-loss-prohibited**, the procedure generates a negative report instruction with the **non-delivery-reason-code conversion-not-performed** and one of the following **non-delivery-diagnostic-codes**, as appropriate:

   – **line-too-long,**
   – **page-split,**
   – **pictorial-symbol-loss,**
   – **punctuation-symbol-loss,**
   – **alphabetical-character-loss, or**
   – **multiple-information-loss.**

   The procedure then terminates.

6) If the required conversion is allowable, cannot be performed by this MTA, but can be performed by an MTA known to this MTA, then no conversion instruction is generated. The routing instruction previously generated is changed to Transfer-out or Probe-out, with a next hop destination appropriate to the MTA in question. The procedure then terminates.

7) If the required conversion can be performed by this MTA, the procedure returns an instruction to perform the conversion and terminates.

14.3.6 *Error-processing procedure*

When another procedure encounters a deliverability or routing error, this procedure is called to determine whether delivery or routing can be achieved by reassignment of the recipient or by choosing a different **OR-address** for the same recipient. If not, non-delivery must be signalled to the Report module. Errors provoking a call on this procedure include:

– **recipient-name** does not identify an MTS-user;
– delivery failure;
– MTA is unable to perform necessary conversion;
– transfer path problems;
– DL-expansion problems;
– security violations;
– conflict with Registration parameters.

Note – The action taken on error-processing shall be subject to the security-policy in force.

14.3.6.1 *Arguments*

1) A message or probe with the per-recipient fields that caused the problem.
2) Report instructions indicating the error.

14.3.6.2 *Results*

The message or probe in question with an updated **recipient-name** field, or

1) the message or probe in question;
2) report instructions.

14.3.6.3 *Errors*

None.

14.3.6.4 *Procedure description*

Note – This procedure may be called multiple times for a given recipient. Eventually all alternatives will be exhausted and step 5 executed to report failure.

1) The arguments are checked for inclusion of a **directory-name**. If present, the procedure performs a Directoy loo-up to determine a new **OR-address. The OR-address**, if any, thus extracted from the Directory is checked for satisfaction of the **requested-delivery-method** argument, if present. If the check succeeds, the new **OR-address** is substituted for the old and the procedure terminates.

   Note – Following the substitution of the new **OR-address** for the original, the message may legitimately be routed to an MD/MTA that it has already visited. The technique used to prevent premature detection of a routing loop is for further study.

2) Otherwise the procedure determinates whether an **originator-requested-alternate-recipient** was specified for the recipient of concern. If so, the Redirection procedure is called with the message, relevant fields indicated, as argument. Upon successful return from Redirection, the procedure terminates, returning the now redirected message as result.

3) Otherwise the procedure checks for a delivery error, and if present checks the error's cause by examination of the **non-delivery-reason-code** and **non-delivery-diagnostic-code**. If the recipient **OR-address** does not identify an MTS-user, then the **per-message-indicators** are checked for **alternate-recipient-allowed**. If the value found is **alternate-recipient-allowed**, and the MTA has been configured with the address of an alternate-recipient for this class of recipient, then Redirection is called to redirect the message to the alternate-recipient. Upon successful return from Redirection, the procedure terminates, returning the now redirected message as result.

4) The handling of errors which can be resolved but are due to other than addressing problems is a local matter, for example routing to another MTA within the domain because of conversion problems.

5) If the delivery error is of a type other than those cited above, or if the value of **alternate-recipient-allowed** is **alternate-recipient-prohibited**, or if no suitable MD-specified alternate-recipient exists, then the procedure returns a report instruction and terminates.

14.3.7 *Redirection procedure*

This procedure redirects a message to an alternate-recipient.

Note – The use of redirection facilities shall be subject to the security-policy in force.

14.3.7.1 *Arguments*

1) The **OR-name** of the alternate-recipient to whom the message is to be redirected.
2) The per-recipient message fields for the recipient to be replaced by an alternate.
3) The message or probe which is to be redirected.
4) The redirection reason.

14.3.7.2 *Results*

The message or probe supplied in the third argument with the recipient identified in the second argument replaced by the alternate-recipient in the first argument.

14.3.7.3 *Errors*

An indication that a redirection loop has been detected.

14.3.7.4 *Procedure description*

1) The procedure first ensures that redirection to the specified alternate recipient would not result in a redirection loop. The **OR-name** of the alternate-recipient supplied in argument 1 is compared with each **intended-recipient-name** from the sequence of **redirection-history** from the per-recipient fields identified in argument 2. Upon a match the procedure terminates indicating that a redirection loop has been detected.

2) An element is appended to the **redirection-history** (which is created if not present), using the **recipient-name** from argument 2 to form the **intented-recipient-name**, obtaining the **redirection-reason** from argument 4 and containing the Time at which this redirection is performed. The **OR-name** supplied in the first argument is then substituted for that **recipient-name**.

3) In the **other-actions** field of the current **trace-information**, the value **redirected** is set to true.

4) The message transfer envelope is updated as follows:

recipient-name:
replaced

trace-information:
indicate **redirected**

redirection-history:
append previous **recipient-name** and **redirection-reason**

originator-requested-alternate-recipient:
deleted if, and only if the **redirection-reason** indicates **originator-requested-alternate-recipient**

14.3.8 *Splitter procedure*

The splitter replicates messages and probes as required for further processing. The replicas are modified as appropriate to correctly indicate the distribution of responsibility for the various recipients from the original. Each replica is accompanied by a per-message instruction indicating its further disposition within the MTA.

Note – The use of Splitter facilities shall be subject to the security-policy in force.

14.3.8.1 *Arguments*

A message or probe. For each recipient with **responsibility** true a per-recipient routing/conversion instruction accompanies the message.

14.3.8.2 *Results*

One or more replicas of the original message or probe with responsibility appropriately indicated, and a per-message instruction indicating the replica's further disposition within the MTA.

14.3.8.3 *Errors*

None.

14.3.8.4 *Procedure description*

The splitter examines the instructions generated by the Routing-and-conversion-decision procedure to (conceptually) segregate the recipients with **responsibility** true into groups. A replica is created for each group. Further processing for that replica (in other procedures) is dependent on the routing and conversion instructions applicable to the group it represents.

Note 1 – Message replication is required in an MTA because of the potentially differing treatment required for a message's various recipients. These differences arise from the need for more than one relaying path outward from an MTA, from the need for more than one conversion to be carried out on the message's content and from the need to expand distribution lists. For example when more than one relay path exists, a separate copy of the message must be created for each such path, with **responsibility** values as appropriate for the recipients lying along that path.

Note 2 – The determination of what replicas are needed is a local matter, undertaken to minimize the total number of such replicas created. The following paragraphs suggest one approach but are not intended to constrain in any way the approach followed in an actual implementation.

Note 3 – For simplicity of exposition, the Splitter is described as a single-pass algorithm. That is, all necessary replicas are created prior to any further processing. An important optimization would be to minimally split the message for conversion, and then to complete the splitting of the converted copies.

1) The procedure considers first those recipients for which content conversion instructions exist. These recipients are grouped such that the members of each group are subject to identical conversion instructions. A replica is created for each such group with **responsibility** true for the recipients in that group, false for all others.
2) The recipients are then examined for those for which DL-expansion instructions exist. A replica is created for each such DL recipient with **responsibility** false for all recipients but the single DL that yielded the replica.
3) The groups are further subdivided based on per-recipient routing instruction calls for Transfer-out or Probe-out. These recipients are grouped such that each group shares a common next hop destination. A replica is created for each such group with **responsibility** true for recipients in the group, false for all others. For all recipients in each such group, this will be either the first relay attempt of a rerouting attempt. In the latter case the trace-information for the message or probe is modified to indicate that this is a first or subsequent rerouting.
4) Finally, the routing instructions for some recipients will call for Message-delivery or Report-generation. A replica is created for each such subgroup with **responsibility** true for the recipients in the group, false for all others.
5) The procedure now terminates.

14.3.9 *Conversion-procedure*

This procedure performs conversions on messages and indicates those conversions that would have been performed on probes.

14.3.9.1 *Arguments*

A message or probe with the required conversion(s) indicated.

14.3.9.2 *Results*

The message or probe with conversions performed and indicated (just indicated in the case of a probe).

14.3.9.3 *Errors*

The message or probe with report instructions detailing the conversion problem encountered.

14.3.9.4 *Procedure description*

1) For a message, the conversion procedures for built in EITs are performed as defined in Recommendation X,408. The conversion procedures between externally defined EITs and between built in and externally defined EITs are outside the scope of this Recommendation.
2) Upon conversion the message or probe's **trace-information** for this domain is updated to show the converted EITs. The procedure now terminates.

14.3.10 *Distribution-list-expansion procedure*

This procedure takes a message with a single DL recipient and returns a message whose recipient list includes the members of the DL. For a probe it verifies whether DL-expansion would occur, if requested.

Note – The use of DL-expansion shall be subject to the security-policy in force.

14.3.10.1 *Arguments*

1) A message with information indicating the recipient DL which is to be expanded, or
2) a probe with information indicating the recipient DL whose expansion is to be verified.

14.3.10.2 *Results*

1) The message with zero or more recipients representing the DL's membership. Other fields can be updated as indicated in the procedure description below;

2) optionally, the message with report generation instructions to indicate successful delivery,

3) the probe with a report generation instruction.

14.3.10.3 *Errors*

1) A report instruction indicating delivery failure. Values for the **non-delivery-reason-code** and **non-delivery-diagnostic-code** are as indicated in the procedure description below.

2) In the case of DL recursion the procedure terminates without returning errors or results.

14.3.10.4 *Procedure description*

1) For a message (not a probe), do Recursion Detection: The components of the **DL-expansion-history** field are examined for an occurrence of the DL recipient's name. Note that a distinguished **OR-name** of the DL is used for recursion detection, and each expansion point is responsible for ensuring that only that **OR-name** is placed in the **DL-expansion-history**.

   If the DL recipients name is present in the **DL-expansion-history**, then the DL is recursively defined and shall not be expanded further. The message is discarded and no reports or other results are returned. The expansion procedure terminates.

2) DL acquisition: The expansion procedure attempts to acquire the DL attributes.

   If unsuccessful the procedure returns a report instruction with the **non-delivery-reason-code-unable-to-transfer** and **non-delivery-diagnostic-code** as appropriate. The procedure then terminates.

3) Submit permission verification: If it is a message (not a Probe), the last element of the **DL-expansion-history** field (if present) else the **originator-name** is considered to be the sender of the message. For a probe the originator is the sender of the message.

   The sender's name is compared against the components of the DL-submit-permission. If no match, return a report instruction with the **non-delivery-reason-code unable-to-transfer** and **non-delivery-diagnostic-code no-DL-submit-permission**. The procedure then terminates.

4) For a probe: If no other local policy would prevent an attempted delivery, then return a report instruction for successful delivery indication. Procedure then terminates.

5) For a message: The DL recipient's **responsibility** flag is set to false and the DL's members are added as new recipients of the message. The per-recipient fields for each new recipient are copied from that of the DL recipient, except as follows:

   – **recipient-name**: member of the DL.

   The following per-recipient fields are copied or changed according to local DL policy:

   – **DL-expansion-prohibited**,

   – **originating-MTA-report-request** (see Note 1),

   – **originator-report-request** (see Note 1),

   – **originator-requested-alternate-recipient** (see Note 2),

   – **explicit-conversion**.

   Note 1 – Copy only if DL-policy requires and the originator would not receive unrequested reports.

   Note 2 – The **originator-requested-alternate-recipient** can be removed, or replaced, according to local DL policy, or copied, but only if explicity required by DL policy.

   Note 3 – Any DL-members that identify DLs that are already present in the **DL-expansion-history** may be excluded from the DL expansion and not included in the new recipients of the message.

6) In the **other-actions** field of the current **trace-information**, the value **dl-operation** is set to true.

7) The distinguished value of the DL's **OR-name** (including its **OR-address**) and the Time at which this expansion occurred are appended to the **DL-expansion-history** field of the message.

Note – The use of a distinguished value of the DL's **OR-name** here refers not to distinguished **directory-name** but to a specific **OR-name** of the DL which the expansion point chooses to use for comparison purposes.

8) If the new report request values (determined in step 5) or the DL's local policy will prevent the originator from receiving a requested delivery report from the DL's members, then a copy of the message, with delivery report request instructions for the expanded DL, is constructed and returned along with the message.

9) The procedure returns the revised message and the optional report request and then terminates.

14.3.11 *Loop detection and routing algorithm*

The routing and loop detection algorithms for inter or intra domain use are beyond the scope of this Recommendation. In order to expose the issues that must be considered, the remainder of this clause describes one approach toward routing and loop detection. This material is not part of the Recommendation.

The paragraphs that follow describe a simple method of loop detection together with a minimal routing algorithm. The algorithm is minimal in the sense that it presupposes only minimal knowledge from each MD and performs transfer steps that avoid loops (in the sense indicated below). Of course, this algorithm can be improved any time an MD knows more about the topology of the network of MDs.

The algorithm recognizes the fact that it is in general legitimate (i.e. no loop should be detected) to re-enter an MD if a specific operation has been performed by another MD since the last passage through the MD about to be re-entered. Legitimate operations are: conversion, DL-expansion, and redirection.

1) Notation: The Trace Information sequence is made of **trace-information-elements** denoted in a simplified way as [MD, routing-action, operation], where MD is the name of an MD; routing-action is "relayed" or "rerouted", operation is "conversion", "DL-operation", "redirection" or "nil". M denotes the message to transfer. MD(o) denotes the current MD (the one currently doing loop detection). Neighbours is the set of selected adjacent MDs (neighbours of MD(o)), which are possible relay-MDs for M. Trace-Info* is the suffix of Trace-Info obtained by considering the tail of the trace info sequence beginning with the last [MD, r, op] trace info element where op is not nil (nil indicates that no operation has been performed by an MD).

2) Loop Detection: Examine Trace-Info for loops. A loop is detected if the trace info sequence contains a suffix, [MD(o), relayed, op(o)] ... [MD(p), relayed, op(p)] where for all j of which $o < j \leqslant p$ the associated trace info element is [MD(j), relayed, op(j)] and op(j) = nil. That is, a loop is detected if M arrives at an MD which has already relayed it and each MD afterwards has also relayed it without performing any operation other than routing. If a loop is detected, then the algorithm returns an error indicating the problem, and terminates.

3) Routing Setup: If no loop is detected, the set, Neighbours, is adjusted, if necessary, for loop-avoiding transfer steps in the context of the current message. (The adjustment affects other message.)

   a) If there is no loop and no occurrence of [MD(o), r, op], in Trace-Info*, then Neighbours is unchanged.

   b) If there is no loop but there is an occurrence of [MD(o), r, op] in Trace-Info*, then remove from Neighbours all MDs which appear in that suffix of Trace-Info* which begins with [MD(o), r, op]. Modify the trace info element added by the current domain to show rerouted as routing action. Add a previous-MD parameter determined as follows: The last [MD(o), r, op] trace info element in Trace Info is located. The previous-MD is the MD appearing in the first trace info element after this last [MD(o), r, op] trace info element.

   c) In cases a and b, if Neighbours is empty, the algorithm returns an error indicating the problem and terminates.

4) Routing action. A next hop is selected from Neighbours for each recipient to be relayed.

14.4 *Report module*

The Report module can be invoked by:

1) the Report-in module, which passes a report, or
2) the Main module, which passes a message or probe with report instructions;
3) the Report-out module, which passes a report with failure description.

If an error is encountered by the procedures internal to this module, no output is generated. Otherwise the Report module invokes the Report-out or Report-delivery module, passing a report with transfer or delivery instructions, respectively. See Figure 10/X.411.

Note – The use of reports shall be subject to the security-policy in force.

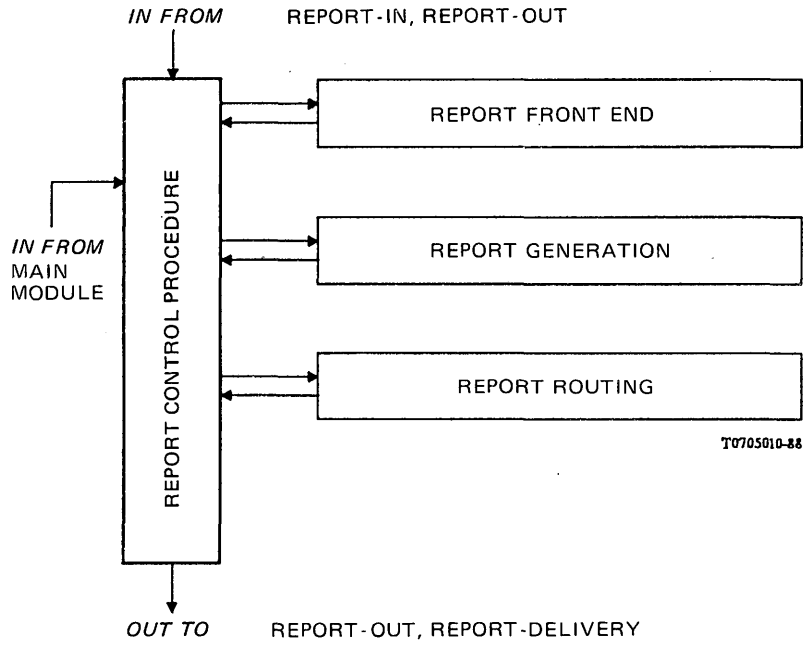


FIGURE 10/X.411

Organization of procedures within the report module

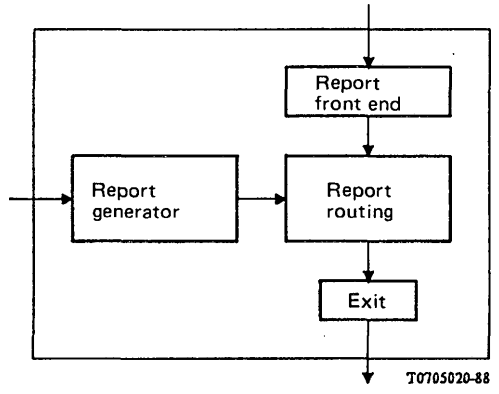


FIGURE 11/X.411

Information flow within the report module

14.4.1 *Control procedure*

14.4.1.1 *Arguments*

1) A report, or
2) a message or probe with report instructions.

14.4.1.2 *Results*

1) A report with relaying or delivery instructions, or
2) no result in case an error is encountered.

14.4.1.3 *Errors*

None. The report, message, or probe is discaded if an error is encountered.

14.4.1.4 *Procedure description*

1) For a report from report-in the report-front-end procedure is first called to perform trace initialization and several initial verification steps. A null return indicates an error; the report is discarded and processing terminates. Otherwise processing continues as step 3 below.
2) For a message or probe the Report-generation procedure is first called to create a report. A null return indicates an error; the message or probe is discarded and processing terminates. If a report is returned, processing continues at step 3, below.
3) The Report-routing procedure is called to generate a routing instruction for the report. A null return indicates an error; the report is discarded and processing terminates. In the case of a positive return the trace update procedure is now called to indicate passage through this MTA. The Control procedure returns the completed report together with routing instruction and terminates, subject to the security-policy.

14.4.2 *Report-front-end procedure*

This procedure performs trace initialization detection of message-expiration violations, initial security check, loop detection and criticality check.

14.4.2.1 *Arguments*

A report.

14.4.2.2 *Results*

The report with initialized **trace-information** for this MTA.

14.4.2.3 *Errors*

None. The report is discarded if an error is detected.

14.4.2.4 *Procedure description*

1) If the report has crossed a domain boundary, a **trace-information-element** for this domain is added with current time as the **arrival-time** and **relay** as **action**. An **internal-trace-information-element** is also added whether or not the report has crossed a domain boundary.
2) If required by the security-policy in force and/or if the **report-origin-authentication-check** is incorrect, the report is discarded and processing terminates.
3) If any of the extension fields is marked critical for transfer but is not semantically understood by the MTA, the report is discarded. The procedure then terminates.
4) Loop detection is performed. The loop detection algorithm is beyond the scope of this Recommendation. However, an example of a combined routing and loop detection algorithm is given in § 14.3.11. If a loop is detected, the report is discarded and the procedure terminates.

14.4.3 *Report-generation procedure*

This procedure generates a report describing the success and/or failure of operations attempted by this MTA.

14.4.3.1 *Arguments*

A message or probe. For each recipient with **responsibility** true, a per-recipient instruction is included indicating the success or problem to be reported.

14.4.3.2 *Results*

A report describing the successes or failures to be reported.

14.4.3.3 *Errors*

None.

14.4.3.4 *Procedure description*

If the subject's **originating-MTA-report-request** field so indicates, the report is constructed with arguments as described in Table 31/X.411, and further amplified by the following:

The delivery arguments (**message-delivery-time**, **type-of-MTS-user**) or Non-delivery arguments (**non-delivery-reason-code**, **non-delivery-diagnostic-code**) for each recipient are taken from the per-recipient instructions that accompanied the subject message. **Message-delivery-time** is taken from the message or probe trace information in case of a delivery report. If failure is reported for a DL recipient, then the **type-of-MTS-user** is set to **DL**. The **report-destination-name** is the last element from **DL-expansion-history**, if that element exists. For messages with no **DL-expansion-history** and for all probes, the **report-destination-name** is the subject's **originator-name**. The **originator-and-DL-expansion** will contain the **originator-name** and the subject's **Message-submission-time** followed by the content of **DL-expansion-history**.

Note – Reporting-DL-name is not generated under any of these conditions.

In the case where the instructions reflect multiple failures, the report should reflect the original problem rather than the failure of subsequent recovery actions.

Note – That the MTA nominates **critically** values for fields copied from the subject. These new values reflect criticality with regard to the report, not the subject. The MTA will not copy into the report any critical functions which it does not support.

14.4.4 *Report-routing procedure*

This procedure determines the routing action, if any, to be taken on a report. Report-routing reflects special conditions that require a routing procedure different from that applicable to messages or probes:

1) A report has just one recipient – the originator of the message that forms the subject of the report, a DL expansion-point, or, if local policy allows, a DL owner.
2) Insurmountable failures encountered in routing a report result in the discarding of the report. No attempt is made to generate a further report on the difficulty encountered.

The processing actions necessitated by these conditions are described in the following clauses. It should be noted that the routing of reports is subject to the security-policy.

14.4.4.1 *Arguments*

One of the following:

1) a report transferred to this MTA from another MTA and successfully processed by the report-front-end procedure;
2) a report created by the Report-generation procedure internal to this MTA;
3) a report received back from the Report-out procedure together with a description of the transfer failure encountered.

14.4.4.2 *Results*

One of the following:

1) the report, together with relaying instructions to the next hop MTA;
2) the report, together with an indication of the locally supported MTS-user who is to receive Report-delivery.

14.4.4.3 *Errors*

None. If no local recipient or next hop can be determined, the report is discarded.

14.4.4.4 *Procedure description*

1) Reports relayed to this MTA or generated locally receive normal routing attention as follows:

   a) If the Report-destination is not local to this MTA then relaying is required. Report-routing attempts to determine the next hop address. In this determination the **message-security-label** of the report is checked against the **security-context** to ensure no violation of the security-policy occurs. If successful, then the report, together with this information is returned as the procedure's result. The procedure then terminates. The report is subsequently passed to the Report-out procedure.

      If the next hop address cannot be determined, then the report is discarded and the procedure terminates without returning a result.

   b) If the Report-destination is an MTS-user local to this MTA, and the **originator-report-request** field indicates, then Report-delivery is required (subject to the security-policy in force). Report-routing attempts to determine the OR-address of the report destination. If successful, then the report, together with this information is returned as the procedure's result. The procedure then terminates. The report is subsequently passed to the Report-delivery procedures.

      If the report was not requested or the report destination address cannot be determined, the report is discarded and the procedure terminates without returning a result.

   c) If the **report-destination-name** is of a DL local to this MTA, then this report is in process of routing back along a path of successive DL expansion-points. In the **other-actions** field of the current **trace-information-element**, the value **dl-expansion** is set to true.

      Any processing based on local DL policy would occur here; e.g. a copy of the report can be constructed and sent to the DL owner. In this case the **report-destination-name** will be that of the DL owner and the **reporting-DL-name** will be constructed to contain the subject DL name. This copy of the report shall not contain the **returned-content**. In addition, suppression of reports can be done here.

      Note – The possibility that a DL owner is itself a DL is for further study.

      If the report is not to be suppressed, the MTA then replaces the **OR-name** currently in the **report-destination-name** field by the OR-name immediately preceding that one in the **originator-and-DL-expansion-history** field. Thus the report acquires, as a new destination, the next entry back along the chain of entries in the **originator-and-DL-expansion-history** field:

      report-destination-name:
      Copy previous DL **OR-name** from originator-and-**DL-expansion-history**.

      reporting-DL-name:
      Generated only in case of reports to DL owner.

      In order to route the report to this new destination, the Report-routing procedure now calls itself recursively. The result returned, if any, from this recursive call is returned, and the procedure terminates.

2) A report received back from the Report-out procedure has encountered a transfer failure in the process of relaying to another MTA. The Report-routing procedure attempts to reroute such a report, i.e. compute an alternative next hop address (subject to the security-policy in force). If an alternative next hop address is found then the report, together with this information and suitably modified trace information is returned as the procedure's result. The procedure then terminates. The report is subsequently passed to the report-out procedures.

   If an alternative next hop address cannot be determined, then the report is discarded and the procedure terminates without returning a result.

14.5 *MTS-bind and MTS-unbind*

14.5.1 *MTS-user initiated MTS-bind procedure*

This paragraph describes the behaviour of the MTA when an MTS-bind is invoked by an MTS-user.

14.5.1.1 *Arguments*

The MTS-bind arguments are defined in § 8.1.1.1.1.

14.5.1.2 *Results*

The MTS-bind results are defined in § 8.1.1.1.2.

14.5.1.3 *Errors*

The bind-errors are defined in § 8.1.2.

14.5.1.4 *Procedure description*

1) If the MTAs resources cannot currently support the establishment of a new association, the procedure returns a Busy bind-error and terminates.
2) Otherwise, if authentication is required by the security-policy, the MTA attempts to both authenticate the MTS-user via the **initiator-credentials** supplied and check the acceptability of the **security-context**. If the **initiator-credentials** cannot be authenticated, the procedure returns an authentication-error and terminates. If the **security-context** is not acceptable, the procedure returns an unacceptable-security-context bind-error and terminates.
3) If authentication is successful and the **security-context** is acceptable then the MTA accepts the requested association. The procedure returns the **MTA-name** and **responder-credentials**. Messages-waiting is also returned if the MTS-user subscribes to the Hold for Delivery element-of-service. The procedure then terminates.
4) If authentication is not required, Messages-waiting is returned if the **MTS-user** subscribes to the Hold for Delivery element-of-service and the procedure terminates.

14.5.2 *MTS-user initiated MTS-unbind procedure*

This paragraph describes the behaviour of the MTA when an MTS-unbind is invoked by an MTS-user in order to release an existing association established by the MTS-user.

14.5.2.1 *Arguments*

None.

14.5.2.2 *Results*

The MTS-unbind procedure returns an empty result as an indication of release of the association.

14.5.2.3 *Errors*

None.

14.5.2.4 *Procedure description*

The procedure releases the association, returns an empty result, and terminates.

14.5.3 *MTA initiaed MTS-bind procedure*

This paragraph describes the steps taken by an MTA when tasked to establish an association with an MTS-user.

14.5.3.1 *Arguments*

The MTS-bind arguments are defined in § 8.1.1.1.1.

14.5.3.2 *Results*

An internal identifier for the association established.

14.5.3.3 *Errors*

The procedure returns a failure indication in the event an association could not be established.

14.5.3.4 *Procedure description*

1) The procedure establishes values for the arguments defined in § 8.1.1.1.1. Messages-waiting may be supplied if the MTS-user subscribes to the hold for delivery element-of-service. Values for **initiator-name**, **security-context**, and **initiator-credentials** are taken from internal information.
2) The procedure determines the **user-address** of the MTS-user and attempts to establish an association with the arguments of § 8.1.1.1.1. If unsuccessful a failure indication is returned and the procedure terminates.
3) If successful, the results returned from the MTS-user (defined in § 8.1.1.1.2) are examined. The **responder-name** is checked for correctness and an attempt is made to authenticate the MTS-user via the **responder-credentials** returned. If either check fails, the procedure closes the connection, returns a failure indication, and terminates.
4) If both checks are successful the procedure returns the association identifier and terminates.

14.5.4 *MTA initiated MTS-unbind procedure*

This procedure is called to release an association with an MTS-user.

14.5.4.1 *Arguments*

This internal identifier for the association to be released.

14.5.4.2 *Results*

The MTS-unbind procedure returns an empty result as an indication of release of the association.

14.5.4.3 *Errors*

None.

14.5.4.4 *Procedure description*

The procedure releases the association, returns an empty result, and terminates.

14.6 *Submission port*

14.6.1 *Message-submission procedure*

This paragraph describes the behaviour of the MTA when the Message-submission abstract-operation is invoked by the MTS-user on a submission port.

14.6.1.1 *Arguments*

The Message-submission arguments listed in Table 3/X.411 and described in paragraphs indicated in that table.

14.6.1.2 *Results*

1) The Message-submission results listed in Table 5/X.411 and described in paragraphs indicated in that table are passed back to the MTS-user.
2) The Deferred Delivery module is invoked and passed the submitted message.

14.6.1.3 *Errors*

See § 8.2.1.1.3 for description of the relevant abstract-errors.

14.6.1.4 *Procedure description*

1) Error Checking

The message-submission procedure checks for error conditions. If any is found, the indicated abstract-error is returned. All further processing is terminated. Responsibility for the intended message is not accepted by the MTA.

Errors of particular interest:

a) Security errors. If the message-security-label is not compatible with the security-context or, if required, the message-origin-authentication-check is incorrect, a security-error is generated.

b) Criticality errors. If any of the extension fields is marked **critical-for-submission**, but not semantically understood by the MTA, an unsupported-critical-function-error is returned.

If no errors are encountered at this stage, processing continues at step 2. Additional errors may be encountered in these later processing stages, in which case the MTA takes action as described above.

2) Name Processing

The following procedure applies to **originator-name**, **recipient-name** and **originator-requested-alternate-recipient**, unless otherwise noted.

a) If the **OR-name** contains only a **directory-name**, the MTA attempts to obtain the **OR-address**.

The MTA may use the **requested-delivery-method**, if present, as an indication of which form of **OR-address** the **directory-name** should be mapped to. If a form of **OR-address** appropriate to the **requested-delivery-method**, cannot be found, the recipient-improperly-specified abstract-error is returned by the MTA.

b) If the **OR-name** contains both the **directory-name** and the **OR-address**, their association need not be validated. If the **OR-address** is later found to be invalid, the MTA proceeds as if the **OR-address** was not supplied in the **OR-name**. The procedure described in (a) above is used to obtain the **OR-address**, which, if valid, replaces the supplied **OR-address** in the **OR-name**.

If the obtained **OR-address** is invalid, an abstract-error is returned as described in (a) above.

c) If a **recipient-name** contains an **OR-address** of a form not appropriate to the **requested-delivery-method**, if present, the **recipient-improperly-specified** abstract-error is returned by the MTA.

d) The validation of the **OR-address**, whether passed in the Message-submission argument or obtained by resolving the **directory-name**, has two steps. The first step validates that the purported **OR-address** has the combination of attributes needed for a valid **OR-address** (see § 8.5.5). The second step, which applies only to the **originator-name**, validates that the **OR-address** is, in fact, the **OR-address** of the MTS-user submitting the message.

3) Transfer or Responsibility, Return of Results

If no errors are detected in the above processing, the MTA accepts responsibility for the message and so signifies by returning the Message-submission results to the MTS-user. The Message-submission results are described in § 8.2.1.1.2. The **message-submission-identifier** and **message-submission-time** arguments are constructed as appropriate by the MTA. The **content-identifier** is identical to the corresponding Message-submission argument. If requested by the originator, the originating-MTA generates the **proof-of-submission** using the algorithm identified by the **proof-of-submission-algorithm-identifier** and the arguments defined in § 8.2.1.1.2.4. In addition the **originating-MTA-certificate** is returned.

4) Message Construction

A Message is constructed from the Message-submission arguments, as possibly modified in the above processing steps, plus additional arguments supplied by the MTA, as specified in § 12.2.1.1.

When complete, the Message-submission procedure terminates and the message is passed to the Deferred Delivery module for further processing.

14.6.2 *Probe-submission procedure*

This paragraph describes the behaviour of the MTA when the Probe-submission abstract-operation is invoked by the MTS-user on a submission-part.

14.6.2.1 *Arguments*

The Probe-submission arguments listed in Table 7/X.411 and described in paragraphs indicated in that table.

14.6.2.2 *Results*

1) The Probe-submission results listed in Table 8/X.411 and described in paragraphs indicated in that table are passed back to the MTS-user.

2) The Main module is invoked and passed the submitted probe.

14.6.2.3 *Errors*

See § 8.2.1.2.3 for descriptions of the relevant abstract-errors.

14.6.2.4 *Procedure description*

1) Error Checking

The Probe-submission procedure checks for error conditions. If any is found, the indicated abstract-error is returned. Responsibility for the intended probe is not accepted by the MTA.

Errors of particular interest:

a) Security errors. If the **message-security-label** is not compatible with the **security-context**, or if the **probe-origin-authentication-check** is incorrect, a security-error is generated.

b) Criticality errors. If any of the extension-fields is **critical-for-submission**, but not semantically understood by the MTA, an unsupported-critical-function-error is returned.

If no errors are encountered at this stage, processing continues at step 2. Additional errors may be encountered in these later processing stages, in which case the MTA takes action as described above.

2) Name Processing

The following procedure applies to **originator-name**, **recipient-name** and **originator-requested-alternate-recipient**, unless otherwise noted.

a) If the **OR-name** contains only a **directory-name**, the MTA attempts to obtain the **OR-address**.

In the case of **recipient-name**, the MTA may use the **requested-delivery-method**, if present, to indicate which form of **OR-address** the **directory-name** should be mapped to. If a form of **OR-address** appropriate to the **requested-delivery-method** cannot be found, the recipient-improperly-specified abstract-error is returned to the MTA.

b) If the **OR-name** contains both the **directory-name** and the **OR-address**, their association need not be validated. If the **OR-address** is later found to be invalid, the MTA proceeds as if the **OR-address** was not supplied in the **OR-name**. The procedure described in a) above is used to obtain the **OR-address**, which, if valid, replaces the supplied **OR-address** in the **OR-name**.

If the obtained **OR-address** is invalid, an abstract-error is returned as described in b) above.

c) If a **recipient-name** contains an **OR-address** of a form not appropriate to the **requested-delivery-method**, if present, the recipient-improperly-specified abstract-error is returned by the MTA.

d) The validation of the **OR-address**, whether passed in the Probe-submission argument or obtained by resolving the **directory-name**, has two steps. The first step validates that the purported **OR-address** has the combination of attributes needed for a valid **OR-address** (see § 8.5.5). The second step, which applies only to the **originator-name**, validates that the **OR-address** is, in fact, the **OR-address** of the MTS-user submitting the message.

3) Transfer of Responsibility, Return of Results

If no errors are detected in the above steps, the MTA accepts responsibility for the probe and so signifies by returning the Probe-submission results to the MTS-user. The Probe-submission results are described in § 8.2.1.2.2. The **probe-submission-identifier** and **probe-submission-time** arguments are constructed as appropriate by the MTA. The **content-identifier** is identical to the corresponding Probe-submission argument.

4) Probe Construction

A Probe is constructed from the Probe-submission arguments, as possibly modified in the above processing steps, plus additional arguments supplied by the MTA.

When complete, the Probe-submission procedure terminates and the probe is passed to the main module for further processing.

14.6.3 *Cancel-deferred-delivery procedure*

This paragraph describes the behaviour of the MTA when the Cancel-deferred-delivery abstract-operation is invoked by the MTA-user on a submission-port in order to cancel the deferred delivery message previously submitted to the MTA.

14.6.3.1 *Arguments*

The Cancel-deferred-delivery arguments listed in Table 10/X.411 and described in paragraphs indicated in that table.

14.6.3.2 *Results*

An empty result is passed back to the MTS-user as an indication of successful cancellation.

14.6.3.3 *Errors*

See § 8.2.1.3.3 for descriptions of the relevant abstract-errors.

14.6.3.4 *Procedure description*

1) If a **proof-of-submission** has already been provided, the Too-late-to-cancel abstract-error is returned by the MTA. The deferred delivery of the message is not cancelled.
2) If the value of the **message-submission-identifier** argument is recognized by the MTA as being valid and associated with a message being held by the MTA for deferred-delivery, the MTA discards this message as being cancelled, and assumes no further responsibility for it.
3) If the value of the **message-submission-identifier** argument is recognized by the MTA as being valid but refers to a message already delivered or transferred to another MTA, the Too-late-to-cancel abstract-error is invoked by the MTA. The deferred delivery of the message is not cancelled.
4) If the value of the **message-submission-identifier** argument is not recognized as being valid (either because the MTA never assigned such a value or because the MTA no longer holds the historical record of a deferred delivery message that has been transferred or delivered), then the Message-submission-identifier-invalid or Too-late-to-cancel abstract-error is returned by the MTA, the choice of which being a local matter.

14.6.4 *Submission-control procedure*

This paragraph describes the behaviour of the MTA when invoking the Submission-control abstract-operation on a submission-port in order to temporarily limit the submission-port abstract-operations that the MTS-user can invoke. These controls remain in force for the duration of the current association unless overridden by a subsequent Submission-control abstract-operation.

Note – The use of Submission-control shall be subject to the security-policy in force. The **permissible-security-context** Submission-control argument limits the **security-context** established during the MTS-bind.

14.6.4.1 *Arguments*

The Submission-control arguments listed in Table 12/X.411 and described in paragraphs indicated in that table.

14.6.4.2 *Results*

The Submission-control results listed in Table 13/X.411 and described in paragraphs indicated in that table are passed back to the MTA by the MTS-user.

14.6.4.3 *Errors*

A Security-error can be passed back by the MTS-user. See § 8.2.1.4.3 for a description of this abstract-error.

14.6.4.4 *Procedure description*

The circumstances causing an MTA to invoke the Submission-control abstract-operation are a local matter, as are the actions taken during and subsequent to its completion.

14.7 *Delivery port*

14.7.1 *Message-delivery procedure*

This paragraph describes the steps taken by an MTA when tasked to deliver a message to one or more MTS-users.

Most provisions of this clause also apply to the case where the MTA has received a probe with one or more local recipients. Unless noted otherwise, all procedure steps save physical delivery apply to the handling of probes.

Note – The generation of reports shall be subject to the security-policy.

14.7.1.1 *Arguments*

1) A message from the main module with per-recipient instructions to deliver to one or more local MTS-users.
2) The message-delivery arguments listed in Table 15/X.411 and described in paragraphs indicated in that table are passed to the recipient MTS-user.

14.7.1.2 *Results*

1) An empty or, if requested, a **proof-of-delivery** and optional **recipient-certificate** result passed back from the MTS-user as an indication of successful delivery with no reporting requirements.
2) The Main module is invoked and passed the message with per-recipient instructions describing any delivery problems encountered and/or indicating successful deliveries to be reported on.

14.7.1.3 *Errors*

Message-delivery abstract-errors that can be returned from the MTS-user to the MTA are described in § 8.3.1.1.3. These error conditions are reported to the Main module in the results described above.

14.7.1.4 *Procedure description*

1) If the message expiration is reached, a report instruction is generated for each local recipient. The values of **non-delivery-reason-code** and **non-delivery-diagnostic-code** are **unable-to-transfer** and **maximum-time-expired**, respectively. The procedure then terminates.
2) If any of the per-message **extension-fields** is set to **critical-for-delivery** but not semantically understood by the MTA, a report instruction for each local recipient is generated. The values of **non-delivery-reason-code** and**non-delivery-diagnostic-code** are set to **unable-to-transfer** and **unsupported-critical-function** respectively.
3) Otherwise, values are established for those arguments to the Message-delivery abstract-operation that apply to all recipients (arguments to message-delivery are described in § 8.3.1.1.1).
4) Steps 4-15 are executed for each recipient with **responsibility** true. The procedure then terminates.
5) To ensure the security-policy is not violated during delivery, the **message-security-label** is checked against the **security-context**. If delivery is barred by the security-policy then, subject to the security policy, a report instruction for this recipient is generated. The values of **non-delivery-reason-code** and **non-delivery-diagnostic-code** are **unable-to-transfer** and **secure-messaging-error**, respectively.

6) If delivery barred by restrictions imposed in a previously invoked Register or Delivery-control-abstract-operation, then, subject to the security-policy in force, the MTA will hold the message pending the lifting of the applicable restriction(s).

7) If the maximum holding time for a held message (the value of this maximum time being a local matter) expires with the applicable restrictions still in effect, then a report instruction is generated for this recipient. The values of **non-delivery-reason-code** and **non-delivery-diagnostic-code** are **unable-to-transfer** and **recipient-unavailable**, respectively. Processing then terminates for this recipient.

Note – The processing steps (5 and 6 above) associated with control restrictions do not apply in the case of Probe.

8) If restricted delivery is enforced and the recipient falls in the category of unauthorized senders, then a report instruction is generated for this recipient. The value of **non-delivery-reason-code** is set to **restricted-delivery**. Processing then terminates for this recipient.

9) The MTA establishes those arguments for the Message-delivery abstract-operation that apply only to the individual recipient: **message-delivery-identifier** and **message-delivery-time** are given values as described in §§ 8.3.1.1.1.1 and 8.3.1.1.1.2. All other arguments are taken directly from corresponding fields of the message to be delivered. With the exceptions noted below, all arguments shown in Table 11/X.411 are included in each invocation of Message-delivery.

10) If **disclosure-of-recipients** has the value **disclosure-of recipients-allowed**, the MTA includes all recipients, which were specified by the originator, save the current one, in the **other-recipient-name argument**.

Note that if the recipient is a member of a distribution list, other members of this distribution list must not be included in the **other-recipient-name** argument. The recipient is a member of a distribution list if the **DL-expansion-history** field is non-empty.

11) If any of the per-recipient **extension-fields** is set to **critical-for-delivery**, but not semantically understood by the MTA, a report instruction for this recipient is generated. The values of the **non-delivery-reason-code** and **non-delivery-diagnostic-code** are set to **unable-to-transfer** and **unsupported-critical-function** respectively.

12) In the case of delivery to a Physical Delivery Access Unit, the Physical Delivery Arguments are included in the Message-delivery. These arguments are described in §§ 8.2.1.1.1.14-8.2.1.1.1.23.

13) Once all conditions have been met for succesful delivery, the MTA will physically deliver the message. The accomplishment of delivery to a collocated recipient MTS-user is a local matter. In the case of a remotely located recipient MTS-user, the MTA establishes an association with that MTS-user (or uses an existing one) and invokes the Message-delivery abstract-operation across that association. With successful delivery, either remote or local, responsibility for the message passes from the MTA to the recipient MTS-user.

14) Upon a successful delivery, if the **originating-MTA-delivery-report-request** has the value of **report** or **audited-report**, then a report instruction is generated noting the successful delivery. Processing then terminates for this recipient.

15) In the case of a remotely located recipient MTS-user, if an association neither exists nor can be established initially, or there is a transfer failure across an association, the MTA can repeat the attempt at association establishment and/or transfer, the maximum number and/or time duration of repeats being a local matter. If, after repeated attempts transfer has not been accomplished, the message is deemed undeliverable and, subject to the security-policy in force, a report instruction is generated. The values of **non-delivery-reason-code** and **non-delivery-diagnostic-code** are **transfer-failure** and **recipient-unavailable**, respectively. Processing then terminates for this **recipient**.

Note – The processing steps associated with physical transfer of a message to the recipient MTS-user do not apply in the case of Probe.

16) Return of results and errors by the MTS-user.

If the Message-delivery abstract-operation is successful, then the MTS-user returns, as an indication of success either an empty result or, if requested, a **proof-of-delivery** and optional recipient-certificate.

If the Message-delivery abstract-operation violates one or more controls imposed by a previous Delivery-control or Register abstract-operation, then the MTS-user returns a Delivery-control-violated error. If the **security-context** dictates that the MTS-user cannot support the requested abstract-operation because it would violate the security-policy, then the MTS-user returns a Security-error. In this event the Message-delivery invocation has failed and the MTA retains responsibility for the message with respect to this recipient. The message is held for subsequent retry or is passed to the Main module for report generation. Processing then terminates for this recipient.

14.7.2 *Probe-delivery-test procedure*

This paragraph describes the steps taken by an MTA when tasked to test the deliverability of probe.

Note – The use of Reports shall be subject to the security-policy.

14.7.2.1 *Arguments*

1) A probe from the internal procedure with per-recipient instructions to Probe-delivery-test to one or more local MTS-users.

14.7.2.2 *Results*

The Main module is invoked and passed the probe with per-recipient instructions describing whether or not the hypothetical delivery would have occurred and if not why not.

14.7.2.3 *Errors*

None.

14.7.2.4 *Procedure description*

The logic for Message-delivery is described in § 14.7.1. All steps in the paragraph except those specifically noted as inapplicable to Probe are executed.

14.7.3 *Report-delivery procedure*

This paragraph describes the steps taken by an MTA when tasked to deliver a report to an MTS-user. Report-delivery is called for when an MTA receives a report, from Report-in or upon generation within this MTA, whose **originator-name** field specifies an MTS-user served by this MTA.

14.7.3.1 *Arguments*

1) A report from the Report module with per-recipient instructions to deliver to a local recipient.
2) The Report-delivery arguments listed in Table 18/X.411 and described in paragraphs indicated in that table are passed to the recipient MTS-user.

14.7.3.2 *Results*

An empty result passed back from the MTS-user as an indication of successful delivery.

14.7.3.3 *Errors*

Report-delivery errors that can be returned from the MTS-user to the MTA are described in § 8.3.1.2.3.

14.7.3.4 *Procedure description*

1) To ensure the security-policy is not voilated during Report-delivery the **message-security-label** is checked against the security-context. If Report-delivery is barred by the security-policy, then the report is descarded.
2) If Report delivery is barred by restrictions imposed in a previously invoked Register or Delivery-control abstract-operation, then, subject to the security-policy in force, the MTA will hold the report pending the lifting of the applicable restriction(s). Restrictions are established by arguments of the Delivery-control or Register abstract-operation as described in § 8.3.1.3.1.

   If the maximum holding time for a held report (the value of this maximum time being a local matter) expires with the applicable restrictions still in effect, then the report is discarded.
3) Arguments for the Report-delivery abstract-operation are taken from corresponding fields of the report.
4) If any of the per-message or per-recipient **extension-fields** are set to **critical-for-delivery**, but not semantically understood by the MTA, the report is discarded.

5) The accomplishment of Report-delivery to a collocated MTS-user is a local matter. In the case of a remotely located MTS-user, the MTA establishes an association with that MTS-user (or uses an existing one) and invokes the Report-delivery abstract-operation across that association. With successful Report-delivery, either remote or local, responsibility for the report passes from the MTA to the MTS-user.

6) In the case of a remotely located MTS-user, if an association cannot be established initially, the MTA can repeat the attempt, the maximum number and/or time duration of repeats being a local matter. If, after repeated attempts no association has been established, the report is deemed undeliverable and is discarded.

7) Return of Results and Errors by the MTS-user.

   If the Report-delivery abstract-operation is successful, then the MTS-user returns an empty result as an indication of success.

   If the Report-delivery abstract-operation violates one or more controls imposed by a previous Delivery-control or Register abstract-operation, then the MTS-user returns a Delivery-control-violated error. In this event the Report-delivery invocation has failed and the MTA retains responsibility for the report.

14.7.4 *Delivery-control procedure*

This paragraph describes the behavior of the MTA when the Delivery-control abstract-operation is invoked by an MTS-user served by this MTA. Delivery-control imposes and lifts restrictions on the Message-delivery and Report-delivery abstract-operations. These controls remain in force for the duration of the current association unless overridden by a subsequent Delivery-control. Delivery-controls temporarily limit the **security-context** but cannot cause a violation of the security-policy.

These controls do not apply to the processing of probes by the MTA.

14.7.4.1 *Arguments*

The Delivery-control arguments listed in Table 20/X.411 and described in § 8.3.1.3.1.

14.7.4.2 *Results*

1) The Delivery-control results listed in Table 21/X.411 and described in § 8.3.1.3.2 are passed back to the MTS-user by the MTA.

2) Various control parameters of the MTS-user held by this MTA are replaced by values carried in the Delivery-control arguments.

14.7.4.3 *Errors*

See § 8.3.1.3.3 for a description of the relevant abstract-errors.

14.7.4.4 *Procedure description*

1) If the value of the **restrict** argument is **remove**, then all controls established by any previous Delivery-control are removed; the abstract-operation is complete, and the Result is returned to the MTS-user.

2) If the value of the restrict argument is **update**, and no other arguments are present, the request is considered to be valid and the Result returned to the MTS-user.

   In such cases all currently in force control values remain unchanged.

3) If the value of the **restrict** argument is **update**, and other arguments are present, those arguments are checked for compatibility with long term conditions specified by the most recent invocation of the Register abstract-operation on the administration-port (see § 14.4.1). If no incompatibility is detected, and the update is permitted within the security-policy, the indicated updates are carried out, the abstract-operation is complete, annd the Result is returned to the MTS-user.

4) If any of the following incompatibilities is detected with long term conditions, a Control-violates-registration abstract-error is returned by the MTA;

a) The **permissible-encoded-information-types** has a type not specified among those allowed long term.

b) The **permissible-content-types** has a content not specified among those allowed long term.

c) The **permissible-maximum-content-length** exceeds the length allowed long term.

d) The **permissible-security-context** is violated.

In any of the error cases, the Delivery-control is discarded and not carried out.

14.8 *Administration port*

14.8.1 *Register procedure*

This paragraph describes the behaviour of the MTA when the Register abstract-operation is invoked by an MTS-user served by this MTA.

14.8.1.1 *Arguments*

The Register arguments listed in Table 23/X.411 and described in paragraphs indicated in that table.

14.8.1.2 *Results*

1) The Register procedure returns an empty result to the MTS-user as an indication of success.

2) Various parameters of the MTS-user held by this MTA are replaced by values carried in the Register arguments.

14.8.1.3 *Errors*

A Register-rejected error returned to the MTS-user as described in § 8.4.1.1.3.

14.8.1.4 *Procedure description*

1) The Register arguments are checked for correct specification. If any is incorrectly specified, the Register procedure returns a Register-rejected error and terminates.

2) If the Register arguments are correctly specified, the values of MTS-user parameters are replaced by those of the Register arguments, and the procedure terminates.

14.8.2 *MTS-user initiated change-credentials procedure*

This paragraph describes the behavior of the MTA when a change-credentials abstract-operation is invoked by the MTS-user.

Note – All changes of credentials shall be subject to the security-policy in force.

14.8.2.1 *Arguments*

The Change-credentials arguments listed in Table 25/X.411 and described in § 8.4.1.2.1.

14.8.2.2 *Results*

1) The Change-credentials procedure returns an empty result to the MTS-user as an indication of success.

2) The MTS-user's credentials held by this MTA are changed in accordance with the new-credentials argument.

14.8.2.3 *Errors*

A New-credentials-unacceptable or Old-credentials-incorrectly-specified abstract-error, as described in § 8.4.1.2.3 and listed in Table 26/X.411.

14.8.2.4 *Procedure description*

Note – All changes of credentials shall be subject to the security-policy in force.

1) If the value of the **old-credentials** argument is not the same as the credentials held by the MTA for the MTS-user invoking the abstract-operation, an Old-credentials-incorrectly-specified error is returned to the MTS-user and the Change-credentials procedure terminates.
2) Otherwise, the **new-credentials** argument is checked for validity. If found invalid (a local matter dictated by the security-policy) a New-credentials-unacceptable error is returned to the MTS-user and the Change-credentials procedure terminates.
3) Otherwise, the MTS-user's credentials held by this MTA are changed to the value of the **new-credentials** argument, an empty result is returned to the MTS-user as an indication of success, and the Change-credentials procedure terminates.

14.8.3 *MTA initiated change-credentials procedure*

This paragraph describes the behaviour of an MTA when changing its credentials held by a locally supported MTS-user.

Note – All changes of credentials shall be subject to the security-policy in force.

14.8.3.1 *Arguments*

The Change-credentials arguments listed in Table 25/X.411 and described in § 8.4.1.2.1.

14.8.3.2 *Results*

The MTS-user returns an empty result to the Change-credentials procedure as an indication of success.

14.8.3.3 *Errors*

The MTS-user can return a New-credentials-unacceptable or Old-credentials-incorrectly-specified error, as described in § 8.4.1.2.3 and listed in Table 26/X.411.

14.8.3.4 *Procedure description*

Note – All changes of credentials shall be subject to the security-policy in force.

1) The procedure invokes the Change-credentials abstract-operation to change the MTA's credentials held by a locally supported MTS-user. The conditions causing an MTA to change its credentials are a local matter.
2) If either the New-credentials-unacceptable or Old-credentials-incorrectly-specified error is received back from the MTS-user, then the MTA must assume its credentials have not been changed. Further action can be undertaken as a local matter, after which the procedure terminates.
3) If an emply result is received back from the MTS-user, the MTA may assume the procedure has been successful and its credentials changed. The procedure terminates.

14.9 *MTA-bind and MTA-unbind*

14.9.1 *MTA-bind-in procedure*

This paragraph describes the behaviour of the MTA when an MTA-bind is invoked by another MTA.

14.9.1.1 *Arguments*

The MTA-bind results are defined in § 12.1.1.1.1 and listed in Table 27/X.411.

14.9.1.2 *Results*

The MTA-bind results are defined in § 12.1.1.1.2 and listed in Table 28/X.411.

14.9.1.3 *Errors*

The bind-errors are defined in § 12.1.2.

14.9.1.4 *Procedure description*

1) If the MTA's resources cannot currently support the establishment of a new association, the procedure returns a Busy bind-error and terminates.
2) Otherwise, if authentication is required by the security-policy, the MTA attempts to both authenticate the calling MTA via the **initiator-credentials** supplied and check the acceptability of the **security-context**. If the **initiator-credentials** cannot be authenticated, the procedure returns an authentication-error and terminates. If the **security-context** is not acceptable, the procedure returns an unacceptable-security-context error and terminates.
3) If authentication is successful and the **security-context** is acceptable, then the MTA establishes the requested association. The procedure returns the **MTA-name** and **responder-credentials**. The procedure then terminates.
4) If authentication is not required, there are no results to return and the procedure terminates.

14.9.2 *MTA-unbind-in procedure*

This paragraph describes the behaviour of the MTA when an MTA-unbind is invoked by another MTA in order to release an existing association.

14.9.2.1 *Arguments*

None.

14.9.2.2 *Results*

The MTA-unbind-in procedure returns an empty result as an indication of release of the association.

14.9.2.3 *Errors*

None.

14.9.2.4 *Procedure description*

The procedure releases the association, returns an empty result, and terminates.

14.9.3 *MTA-bind-out procedure*

This paragraph describes the steps taken by an MTA when tasked to establish an association with another MTA.

14.9.3.1 *Arguments*

1) The **MTA-name** of the MTA with which the association is to be established.
2) The **security-context** for the association.

14.9.3.2 *Results*

An internal identifier for the association established.

14.9.3.3 *Errors*

The procedure returns a failure indication in the event an association could not be established.

14.9.3.4 *Procedure description*

1) The procedure establishes values for the **arguments** defined in § 12.1.1.1.1. Values for **initiator-name**, **security-context**, and **initiator-credentials** are taken from internal information.
2) The procedure determines the address of the MTA and attempts to establish an association with the arguments of § 12.1.1.1.1. If unsuccessful a failure indication is returned and the procedure terminates.

3) If successful, the results returned from the called MTA (defined in § 12.1.1.1.2) are examined. The **responder-name** is checked for correctness, an attempt is made to authenticate the MTA via the **responder-credentials** returned. If any of the checks fail, the procedure returns a failure indication to the caller, terminates the association, and terminates.

4) If all checks are successful the procedure returns the association identifier and terminates.

14.9.4 *MTA-unbind-out procedure*

This procedure is called to release an association with another MTA.

14.9.4.1 *Arguments*

The internal identifier for the association to be released.

14.9.4.2 *Results*

The MTA-unbind-out procedure returns an empty result as an indication of release of the association.

14.9.4.3 *Errors*

None.

14.9.4.4 *Procedure description*

The procedure releases the association, returns an empty result, and terminates.

14.10 *Transfer port*

Note – The actions taken on the transfer-port are subject to the security-policy in force.

14.10.1 *Message-in procedure*

This paragraph describes the behaviour of the MTA when a Message-transfer abstract-operation is invoked by another MTA on a transfer-port.

14.10.1.1 *Arguments*

The Message-transfer arguments listed in Table 29/X.411 and described in paragraphs indicated in that table.

14.10.1.2 *Results*

1) The Deferred Delivery module is invoked and passed the message transferred in.

14.10.1.3 *Errors*

None.

14.10.1.4 *Procedure description*

On receipt of a message through the occurrence of a Message-transfer abstract-operation (invoked from a neighbour MTA), the Message-in procedure is invoked. This procedure simply passes the message to the Deferred Delivery module to determine the actions to be taken by this MTA.

Responsibility for the message passes to the receiving-MTA with the successful transfer.

14.10.2 *Probe-in procedure*

This paragraph describes the behavior of the MTA when a Probe-transfer abstract-operation is invoked by another MTA on a transfer-port.

14.10.2.1 *Arguments*

The Probe-transfer arguments listed in Table 30/X.411 and described in paragraphs indicated in that table.

14.10.2.2 *Results*

1) The Report module is invoked and passed the report transferred in.

14.10.2.3 *Errors*

None.

14.10.2.4 *Procedure description*

On receipt of a probe through the occurrence of a Probe-transfert abstract-operation (invoked from a neighbour MTA), the Probe-in procedure is invoked. This procedure simply passes the probe to the Main module to determine the actions to be taken by this MTA.

Responsibility for the probe passes to the receiving-MTA with the successful transfer.

14.10.3 *Report-in procedure*

This paragraph describes the behavior of the MTA when it receives a Report on a transfer-port through thje occurrence of a Report-transfer abstract-operation invoked by another MTA, or when it receives an indication for the generation of a report from an access unit such as a PDAU.

14.10.3.1 *Arguments*

The Report arguments listed in Table 31/X.411 and described in paragraphs indicated in that table.

14.10.3.2 *Results*

1) The Report module is invoked and passed the report transferred in.

14.10.3.3 *Errors*

None.

14.10.3.4 *Procedure description*

On receipt of a report through the occurrence of a Report-transfer abstract-operation (invoked from a neighbour MTA), or on receipt of an indication for a report generation from an access unit such as a PDAU, the Report-in procedure is invoked. This procedure simply passes the report to the Report module to determine the actions to be taken by this MTA.

Responsibility for the report passes to the receiving-MTA with the successful transfer.

14.10.4 *Message-out procedure*

This paragraph describes the steps taken by an MTA when tasked to transfer a message to another MTA.

14.10.4.1 *Arguments*

A message from the internal procedure with routing instructions to transfer to another MTA. The fields of this message form the arguments of the Message-transfer abstract-operation as listed in Table 29/X.411.

14.10.4.2 *Results*

None.

14.10.4.3 *Errors*

In case of transfer failure the Main module is invoked and passed the message with a per-message instruction indicating the failure reason.

14.10.4.4 *Procedure description*

The message to be transferred provides the arguments for the Message-transfer abstract-operation. It should be noted that the message may reflect processing (e.g., content conversion, redirection, distribution list expansion) carried out in this or previous MTAs.

1) To ensure the security-policy is not violated during transfer, the **message-security-label** is checked against the **security-context**. If the transfer is barred by either the security-policy or temporary restrictions, then processing continues at step 3, below.

2) Otherwise, the MTA establishes an association with the receiving-MTA (or uses an existing one) and invokes the Message-transfer abstract-operation across that association. The completion of Message-out indicates that the transfer has successful and that the receiving-MTA now accepts responsibility for the message. The Message-out procedure now terminates.

   If an association neither exists nor can be established initially, or there is a transfer failure across an association, the MTA can repeat the attempt at association establishment and/or transfer, the maximum number and/or time duration of repeats being a local matter.

3) If, after repeated attempts transfer has not been accomplished, or a security violation has been detected in step 1, the message is deemed non transferrable and is returned, with failure reason indicated, to the Main module for possible rerouting or redirection. Responsibility for the message remains with the sending MTA. The Message-out procedure now terminates.

14.10.5 *Probe-out procedure*

This paragraph describes the steps taken by an MTA when tasked to transfer a probe to another MTA.

14.10.5.1 *Arguments*

A probe from the internal procedure with routing instructions to transfer to another MTA. The fields of this probe form the arguments of the probe-transfer abstract-operation as listed in Table 30/X.411.

14.10.5.2 *Results*

None.

14.10.5.3 *Errors*

In case of transfer failure the Main module is invoked and passed the probe with a per-message instruction indicating the failure reason.

14.10.5.4 *Procedure description*

The probe to be transferred provides the arguments for the Probe-transfer abstract-operation. It should be noted that the probe may reflect processing (e.g., redirection) carried out in this or previous MTAs.

1) To ensure the security-policy is not violated during transfer, the **message-security-label** is checked against the **security-context**. If the transfer is barred by either the security-policy or temporary restrictions, then processing continues at step 3, below.

2) The MTA establishes an association with the receiving MTA (or uses an existing one) and invokes the Probe-transfer abstract-operation across that association. The completion of Probe-out indicates that the transfer has been successful and that the receiving-MTA now accepts responsibility for the probe. The Probe-out procedure now terminates.

   If an association neither exists nor can be established initially, or there is a transfer failure across an association, the MTA can repeat the attempt at association establishement and/or transfer, the maximum number and/or time duration of repeats being a local matter.

3) If, after repeated attempts transfer has not been accomplished, or a security violation has been detected in step 1 above, then the probe is deemed non transferrable and is returned, with failure reason indicated, to the Main module for possible rerouting or redirection. Responsibility for the probe remains with the sending MTA. The Probe-out procedure now terminates.

14.10.6 *Report-out procedure*

This paragraph describes the steps taken by an MTA when tasked to transfer a report to another MTA.

14.10.6.1 *Arguments*

A report from the internal procedure with routing instructions to transfer to another MTA. The fields of this report form the arguments of the Report-transfer abstract-operation as listed in Table 31/X.411.

14.10.6.2 *Results*

None.

14.10.6.3 *Errors*

The report, together with the reason for transfer failure, to be passed back to the Report module.

14.10.6.4 *Procedure description*

The report to be transferred provides the arguments for the Report-transfer abstract-operation. It should be noted that the report may reflect processing (e.g., redirection) carried out in this or previous MTAs.

1) To ensure the security-policy is not violated during transfer, the **message-security-label** is checked against the **security-context**. If the transfer is barred by either the security-policy or temporary restrictions, then processing continues at step 3, below.

2) The MTA establishes an association with the receiving MTA (or uses an existing one) and invokes the Report-transfer abstract-operation across that association. The completion of Report-out indicates that the transfer has been successful and the receiving-MTA now accepts responsibility for the report. The Report-out procedure now terminates.

   If an association neither exists nor can be established initially, or there is a transfer failure across an association, the MTA can repeat the attempt at association establishment and/or transfer, the maximum number and/or time duration of repeats being a local matter.

3) If, after repeat attempts transfer has not been accomplished, or a security violation has been detected in step 1 above, then the report is deemed non transferrable and is returned, with failure reason indicated, to the report module for possible rerouting. Responsibility for the report remains with the sending MTA. The Report-out procedure now terminates.

ANNEX A

(to Recommendation X.411)

Reference definition of MTS object identifiers

This Annex defines for reference purposes various object identifiers cited in the ASN.1 modules in the body of this Recommendation. The object identifiers are assigned in Figure A-1/X.411.

All object identifiers this Recommendation assigns are assigned in this annex. The annex is definitive for all but those ASN.1 modules and the Message Transfer System itself. The definitive assignments for the former occur in the modules themselves; other references to them appear in IMPORT clauses. The latter is fixed.

```
MTSObjectIdentifiers { joint-iso-ccitt mhs-motis(6) mts(3) modules(0) object-identifiers(0) }

DEFINITIONS IMPLICIT TAGS ::=

BEGIN

-- Prologue
-- Exports everything


IMPORTS -- nothing --;


-- Message transfer system

id-mts OBJECT IDENTIFIER ::= { joint-iso-ccitt mhs-motis(6) mts(3) }      -- not definitive


-- Categories of object identifiers

id-mod OBJECT IDENTIFIER ::= { id-mts 0 }                                 -- modules
id-ot OBJECT IDENTIFIER ::= { id-mts 1 }                                  -- object types
id-pt OBJECT IDENTIFIER ::= { id-mts 2 }                                  -- port types
id-cont OBJECT IDENTIFIER ::= { id-mts 3 }                                -- content types
id-eit OBJECT IDENTIFIER ::= { id-mts 4 }                                 -- encoded information types
id-att OBJECT IDENTIFIER ::= { id-mts 5 }                                 -- attributes
id-tok OBJECT IDENTIFIER ::= { id-mts 6 }                                 -- token types
id-sa OBJECT IDENTIFIER ::= { id-mts 7 }                                  -- secure agent types


-- Modules

id-mod-object-identifiers OBJECT IDENTIFIER ::= { id-mod 0 }              -- not definitive
id-mod-mts-abstract-service OBJECT IDENTIFIER ::= { id-mod 1 }            -- not definitive
id-mod-mta-abstract-service OBJECT IDENTIFIER ::= { id-mod 2 }            -- not definitive
```

FIGURE A-1/X.411 (Part 1 of 3)

Abstract syntax definition of the MTS object identifiers

```
id-mod-upper-bounds OBJECT IDENTIFIER ::= {id-mod 3}          -- not definitive

-- Object types
id-ot-mts OBJECT IDENTIFIER ::= {id-ot 0}
id-ot-mts-user OBJECT IDENTIFIER ::= {id-ot 1}
id-ot-mta OBJECT IDENTIFIER ::= {id-ot 2}

-- Port types
id-pt-submission OBJECT IDENTIFIER ::= {id-pt 0}
id-pt-delivery OBJECT IDENTIFIER ::= {id-pt 1}
id-pt-administration OBJECT IDENTIFIER ::= {id-pt 2}
id-pt-transfer OBJECT IDENTIFIER ::= {id-pt 3}

-- Content types
id-cont-undefined OBJECT IDENTIFIER ::= {id-cont 0}
id-cont-inner-envelope OBJECT IDENTIFIER ::= {id-cont 1}

-- Encoded information types
id-eit-undefined OBJECT IDENTIFIER ::= {id-eit 0}
id-eit-telex OBJECT IDENTIFIER ::= {id-eit 1}
id-eit-ia5-text OBJECT IDENTIFIER ::= {id-eit 2}
id-eit-g3-facsimile OBJECT IDENTIFIER ::= {id-eit 3}
id-eit-g4-class-1 OBJECT IDENTIFIER ::= {id-eit 4}
id-eit-teletex OBJECT IDENTIFIER ::= {id-eit 5}
d-eit-videotex OBJECT IDENTIFIER ::= {id-eit 6}
```

FIGURE A-1/X.411 (Part 2 of 3)

Abstract syntax definition of the MTS object identifiers

```
id-eit-voice OBJECT IDENTIFIER ::= { id-eit 7 }

id-eit-sfd OBJECT IDENTIFIER ::= { id-eit 8 }

id-eit-mixed-mode OBJECT IDENTIFIER ::= { id-eit 9 }


-- Attributes

id-att-physicalRendition-basic OBJECT IDENTIFIER ::= { id-att 0 }


-- Token types

id-tok-asymmetricToken OBJECT IDENTIFIER ::= { id-tok 0 }


-- Secure agent types

id-sa-ua OBJECT IDENTIFIER ::= { id-sa 0 }

id-sa-ms OBJECT IDENTIFIER ::= { id-sa 1 }


END  -- of MTSObjectIdentifiers
```

FIGURE A-1/X.411 (Part 3 of 3)

Abstract syntax definition of the MTS object identifiers

ANNEX B

(to Recommendation X.411)

Reference definition of MTS parameter upper bounds

This annex defines for reference purposes the upper bounds of various variable length data types whose abstract syntaxes are defined in the ASN.1 modules in the body of this Recommendation. The upper bounds are defined in Figure B-1/X.411.

```
MTSUpperBounds { joint-iso-ccitt mhs-motis(6) mts(3) modules(0) upper-bounds(3) }

DEFINITIONS IMPLICIT TAGS ::=

BEGIN


-- Prologue
-- Exports everything


IMPORTS -- nothing --;


-- Upper bounds

ub-integer-options INTEGER ::= 256

ub-queue-size INTEGER ::= 2147483647                -- the largest integer in 32 bits

ub-content-length INTEGER ::= 2147483647            -- the largest integer in 32 bits

ub-password-length INTEGER ::= 62

ub-bit-options INTEGER ::= 16

ub-content-types INTEGER ::= 1024

ub-tsap-id-length INTEGER ::= 16

ub-recipients INTEGER ::= 32767

ub-content-id-length INTEGER ::= 16

ub-x121-address-length INTEGER ::= 15

ub-mts-user-types INTEGER ::= 256

ub-reason-codes INTEGER ::= 32767

ub-diagnostic-codes INTEGER ::= 32767

ub-supplementary-info-length INTEGER ::= 256

ub-extension-types INTEGER ::= 256
```

FIGURE B-1/X.411 (Part 1 of 3)

Abstract syntax definition of MTS upper bounds

ub-recipient-number-for-advice-length INTEGER ::= 32

ub-content-correlator-length INTEGER ::= 512

ub-redirections INTEGER ::= 512

ub-dl-expansions INTEGER ::= 512

ub-built-in-content-type INTEGER ::= 32767

ub-local-id-length INTEGER ::= 32

ub-mta-name-length INTEGER ::= 32

ub-country-name-numeric-length INTEGER ::= 3

ub-country-name-alpha-length INTEGER ::= 2

ub-domain-name-length INTEGER ::= 16

ub-terminal-id-length INTEGER ::= 24

ub-organization-name-length INTEGER ::= 64

ub-numeric-user-id-length INTEGER ::= 32

ub-surname-length INTEGER ::= 40

ub-given-name-length INTEGER ::= 16

ub-initials-length INTEGER ::= 5

ub-generation-qualifier-length INTEGER ::= 3

ub-organizational-units INTEGER ::= 4

ub-organizational-unit-name-length INTEGER ::= 32

ub-domain-defined-attributes INTEGER ::= 4

ub-domain-defined-attribute-type-length INTEGER ::= 8

ub-domain-defined-attribute-value-length INTEGER ::= 128

FIGURE B-1/X.411 (Part 2 of 3)

Abstract syntax definition of MTS upper bounds

```
ub-extension-attributes INTEGER ::= 256

ub-common-name-length INTEGER ::= 64

ub-pds-name-length INTEGER ::= 16

ub-postal-code-length INTEGER ::= 16

ub-pds-parameter-length INTEGER ::= 30

ub-physical-address-lines INTEGER ::= 6

ub-unformatted-address-length INTEGER ::= 180

ub-e163-4-number-length INTEGER ::= 15

ub-e163-4-sub-address-length INTEGER ::= 40

ub-built-in-encoded-information-types INTEGER ::= 32

ub-teletex-private-use-length INTEGER ::= 128

ub-encoded-information-types INTEGER ::= 1024

ub-security-labels INTEGER ::= 256

ub-labels-and-redirections INTEGER ::= 256

ub-security-problems INTEGER ::= 256

ub-privacy-mark-length INTEGER ::= 128

ub-security-categories INTEGER ::= 64

ub-transfers INTEGER ::= 512

ub-bilateral-info INTEGER ::= 1024

ub-additional-info INTEGER ::= 1024


END  -- of MTSUpperBounds
```

FIGURE B-1/X.411 (Part 3 of 3)

Abstract syntax definition of MTS upper bounds

ANNEX C

(to Recommendation X.411)

Differences between ISO/IEC and CCITT versions

This annex identifies the technical differences between the ISO/IEC and CCITT versions of CCITT Recommendation X.411 and ISO/IEC 10021-4.

They are:

1) In CCITT Recommendation X.411, extension fields are identified by integers. ISO/IEC 10021-4 allows, in addition, the use of object identifiers for extensions within and/or between PRMDs.
2) In CCITT Recommendation X.411, size contraints are applied to a number of protocol fields (see Annex B). In ISO/IEC 10021-4, the actual values of the constraints are not an integral part of the Standard.

Recommendation X.413

MESSAGE HANDLING SYSTEMS: MESSAGE STORE: ABSTRACT-SERVICE DEFINITION[1)]

(Melbourne, 1988)

The establishment in various countries of telematic services and computer-based store-and-forward message services in assocation with public data networks creates a need to produce standards to facilitate international message exchange between subscribers to such services.

The CCITT,

considering

(a) the need for message handling services;

(b) the need to transfer and store messages of different types;

(c) that Recommendation X.200 defines the reference model of open systems interconnection of CCITT applications;

(d) that Recommendations X.208, X.217, X.218, and X.219 provide the foundation for CCITT applications;

(e) that the X.500-series Recommendations specify directory services and systems;

(f) that message handling services and systems are specified in a series of Recommendations: X.400, X.402, X.403, X.407, X.408, X.411, X.413, and X.419;

(g) that interpersonal messaging is specified in Recommendations X.420 and T.330;

unanimously declares

(1) that the message stores abstract-service definition is specified in Section 2;

(2) that the general-attribute-types and the general-auto-action-types are specified in Section 3;

(3) that the procedures for message store and the ports realization are specified in Section 4.

TABLE OF CONTENTS

[1)] Recommendation X.413 and ISO 10021-5 [Information processing systems – Text Communication – MOTIS – Message Store: Abstract-service definition] were developed in close collaboration and are technically aligned, except for the differences noted in Appendix G.

SECTION 1 – INTRODUCTION

0 Introduction

This Recommendation is one of a series of Recommendations defining Message Handling (MH) in a distributed open systems environment.

Message Handling provides for the exchange of messages between users on a store-and-forward basis. A message submitted by one user (the originator) is transferred through the message-transfer-system (MTS) and delivered to one or more other users (the recipients).

This Recommendation defines the message store abstract-service (MS abstract-service) which supports message-retrieval from a message store (MS) and indirect-message-submission through the MS in a message handling system (MHS). The MS abstract-service also provides message-administration services, as defined by the message transfer system (MTS) abstract-service.

This Recommendation has been produced by joint CCITT-ISO agreement. The corresponding International Standard is ISO 10021-5. Annex G list the differences between the two documents.

1 Scope

This Recommendation defines the message store abstract-service. This abstract-service is provided by the message store access-protocol (specified in Recommendation X.419) in conjunction with the MTS abstract-service (defined in Recommendations X.411), together with the Remote Operations Service Element (ROSE) services (defined in Recommendation X.219). The abstract-syntax-notation for the application-layer protocols used in this Recommendation is defined in Recommendation X.208.

Other Recommendations define other aspects of the MHS. Recommendation X.400 defines the user-oriented services provided by the MHS. Recommendation X.402 provides an architectural overview of the MHS. Recommenation X.407 provides a description of the abstract-service definition conventions used in MHS. Recommendation X.420 defines the abstract-service for interpersonal messaging and defines the format of interpersonal-messages.

Section 2 of this Recommendation contains the message store abstract-service definition. Paragraph 6 describes the MS model. Paragraph 7 specifies the abstract-syntax-notation for the abstract-bind and the abstract-unbind-operations. Paragraph 8 specifies the abstract-syntax-notation for the operations of the abstract-service. Paragraph 9 specifies the abstract-syntax-notation for the errors of the abstract-service.

Section 3 of this Recommendation defines the general-attribute-types and general-auto-action-types related to the MS. Paragraph 10 contains an overview. Paragraph 11 specifies the abstract-syntax-notation for the general-attribute-types. Paragraph 12 specifies the abstract-syntax-notation for the general-auto-action-types.

Section 4 of this Recommendation describes the procedures for message store and the ports realization. Paragraph 13 contains an overview. Paragraph 14 describes how the message store abstract-service is supplied. Paragraph 15 describes how the message transfer system abstract-service is consumed. Paragraph 16 describes how the MS ports are realized.

No requirement is made for conformance to this Recommendation.

2 References

For a list of references refer to Recommendation X.402.

3 Definitions

3.1 *Common definitions for MHS*

For a list of the common definitions for MHS refer to Recommendation X.402.

3.2 *Message store definitions*

For the purpose of this Recommendation the following definitions apply:

3.2.1 **abstract-association:** An abstract binding between two communication partners, in this Recommendation the binding between a UA and an MS for the provision of the MS abstract-service, or between an MS and an MTA for the provision of the MTS abstract-service.

3.2.2 **abstract-bind-parameters:** Parameters defined in this document which are contained in the abstract-bind operations.

3.2.3 **abstract-unbind-parameters:** Parameters defined in this document which are contained in the abstract-unbind operation.

3.2.4 **administration port:** The port offering the administration (for MTS) set of abstract-service within the MS abstract-service.

3.2.5 **alert abstract-operation:** An abstract-operation which allows the MS to signal, based on selection criteria, to the UA that messages or reports are waiting in the MS. Can only be issued on an existing abstract-association.

3.2.6 **attribute:** The information of a particular type appearing in an entry in an information-base.

3.2.7 **attribute-type:** That component of an attribute which indicates the class of information given by that attribute.

3.2.8 **attribute-value:** A particular instance of that class of information indicated by an attribute type.

3.2.9 **attribute-value-assertion:** A proposition, which may be true, false, or undefined, concerning the values of attributes in an entry.

3.2.10 **auto-action:** Actions, that can be performed automatically by the MS, based on previously registered information from the MS-owner via the UA.

3.2.11 **auto-action-type:** An auto-action-type is used to indicate the type of auto-action, e.g. alert.

3.2.12 **auto-alert:** Auto-alert is the auto-action within the MS, which triggers an alert abstract-operation or another action by the MS.

3.2.13 **auto-forward:** Auto-forward is the auto-action within the MS, which triggers a message to be auto-forwarded to another recipient (or other recipients) by the MS.

3.2.14 **child-entry:** An entry, other than the main-entry in an information-base. The parent-entry for a child-entry can be either the main-entry or another child-entry, depending on the number of entry levels in each case.

3.2.15 **child-sequence-number:** A sequence-number in a parent-entry pointing to a child-entry. A parent-entry can have more than one child-sequence-number value, depending on the number of child-entries.

3.2.16 **conditional (C) component:** An ASN.1 element which shall be present in an instance of its class as dictated by this Recommendation. See **grade**.

3.2.17 **content-length:** An attribute which gives the length of the content of a delivered-message (or returned-content).

3.2.18 **content-returned:** An attribute which signals that a delivered-report (or a delivered-message) contained a returned content.

3.2.19 **converted EITs:** An attribute identifying the encoded-information-types of the message content after conversion.

3.2.20 **creation-time**: An attribute which gives the creation-time (by the MS) of an entry.

3.2.21 **delete abstract-operation**: An abstract-operation used to delete one or more entries from an information-base.

3.2.22 **delivered-EITs**: A multi-valued attribute, giving information about EITs in a delivered-message.

3.2.23 **delivered-message entry**: An entry in the stored-messages information-base resulting from a delivered-message.

3.2.24 **delivered-report entry**: An entry in the stored-messages information-base resulting from a delivered-report.

3.2.25 **entry**: An information set in an information-base. See main-entry and child-entry for further classification of entries.

3.2.26 **entry-information**: A parameter, used in abstract-operations, which conveys selected information from an entry.

3.2.27 **entry-information-selection**: A parameter, used in abstract-operations, which indicates what information from an entry is being requested.

3.2.28 **entry-status**: An attribute giving information about the processing status of that entry. Possible values are new, listed or processed.

3.2.29 **entry-type**: An attribute which signals an entry is associated with a delivered-message or a delivered-report.

3.2.30 **fetch abstract-operation**: An abstract-operation which allows one entry to be fetched from the stored-messages information-base.

3.2.31 **fetch-restrictions**: Restrictions, imposed by the UA, on what kind of messages it is prepared to receive as a result of fetch. The possible restrictions are on message-length, content-types and EITs.

3.2.32 **filter**: A parameter, used in abstract-operations, to test a particular entry in an information-base and is either satisfied or not by that entry.

3.2.33 **filter-item**: An assertion about the presence or value(s) of an attribute of a particular type in an entry under test. Each such assertion is either true or false.

3.2.34 **forwarding-request**: This is a parameter that may be present in a message-submission abstract-operation, invoked by the UA, to request that a message is forwarded from the MS.

3.2.35 **general-attribute**: A set of MS attributes which are valid for all types of messages and reports, independent of content-type. Only these MS attributes are explicitly defined in this Recommendation.

3.2.36 **general-auto-action**: Auto-actions which are valid for all types of messages and reports, independent of content-type. Only these auto-actions are explicitly defined in this Recommendation.

3.2.37 **Grade**: Defined in Recommendation X.402.

3.2.38 **indirect-submission port**: The port offering the indirect-submission abstract-service within the MS abstract-service. The indirect-submission abstract-service offers the same services as the message-submission abtract-service (from the MTS abstract-service) with the added functionality of forwarding messages residing in the MS.

3.2.39 **information-base:** Objects within the MS which store information relevant to the MS abstract-service, e.g. the stored-messages information-base, which stores the messages and reports that have been delivered into the MS.

3.2.40 **information-base-type:** The type of information-base, e.g. the stored-messages.

3.2.41 **limit:** A component in the selector parameter which identifies the maximum number of selected entries to be returned in the result of an abstract-operation.

3.2.42 **list abstract-operation:** An abstract-operation which allows a selection of entries from an information-base and requested attribute information to be returned for those entries.

3.2.43 **listed:** An entry-status value.

3.2.44 **Macro:** See Recommendation X.208.

3.2.45 **main-entry:** For each successful abstract-operation which creates information-base entries, there is always one main-entry. Further, or more detailed, information resulting from the same abstract-operation can be stored in child-entries.

3.2.46 **mandatory (M) component:** An ASN.1 element which shall always be present in an instance of its class. See **grade**.

3.2.47 **matching:** The process of comparing the value supplied in an attribute-value-assertion with the value of the indicated attribute-type stored in the MS or deciding whether the indicated attribute-type is present.

3.2.48 **message retrieval service element (MRSE):** The application-service-element by means of which a receiving UA effects retrieval of messages from an MS, or any of various related tasks.

3.2.49 **MS:** Message store, also used as a shorter form for "MS abstract-service-provider".

3.2.50 **MS abstract-service:** The set of capabilities that the MS offers to its users by means of its ports.

3.2.51 **MS abstract-service-user:** The user of the MS abstract-service. This is the UA.

3.2.52 **MS abstract-service-provider:** The MS which provides the MS abstract-service.

3.2.53 **MS-user:** A shorter form for "MS abstract-service-user".

3.2.54 **message-submission abstract-operation:** An abstract-operation which allows the UA to submit a message to the MTS via the MS, and/or to forward a message from the MS to the MTS..

3.2.55 **multi-valued attribute:** An attribute which can have several values associated with it.

3.2.56 **new:** An entry-status value.

3.2.57 **optional (O) component:** An ASN.1 element which shall be present in an instance of its class at the discretion of the object (e.g. user) supplying that instance. See **grade**.

3.2.58 **original-EITs:** An attribute identifying the original encoded-information-types of the message content.

3.2.59 **override:** A component of the selector parameter indicating that the previously registered-restrictions for this abstract-operation should not apply for this instance of this abstract-operation.

3.2.60 **parent-entry:** A parent-entry has one or more child-entries, which were created as a result of the same abstract-operation. If a parent-entry is not a child-entry of another parent-entry, it is a main entry.

3.2.61 **parent-sequence-number:** A sequence-number in a child-entry poiting to its parent-entry. There can only be one parent-sequence-number in a child-entry.

3.2.62 **partial-attribute-request:** A component of the entry-information-selection which enables the return of only selected values of a multi-valued attribute.

3.2.63 **position:** Positions are parameters used to specify a bound of a range.

3.2.64 **processed:** An entry-status value.

3.2.65 **range:** A parameter, used in abstract-operations, to select a contiguous sequence of entries from an information-base.

3.2.66 **register-MS abstract-operation:** An abstract-operation which allows the UA to register certain information, relevant to the UA-MS interworking, in the MS.

3.2.67 **registration:** Information which is registered in the MS and stored (until changed by the Register-MS abstract-operation) between abstract-associations. (See Register-MS).

3.2.68 **registration-identifier:** An identifier for one particular set of registration-parameters for an auto-action-type.

3.2.69 **retrieval port:** The port offering the retrieval set of abstract-services within the MS abstract-service.

3.2.70 **returned-content entry:** An entry-type in the stored-messages information-base which contains the returned content from a previously submitted message.

3.2.71 **selector:** A parameter, used in abstract-operations, to select entries from an information-base.

3.2.72 **sequence-number:** An attribute which uniquely identifies an entry. Sequence-numbers are allocated in ascending order.

3.2.73 **single-valued attribute:** An attribute which can only have one value associated with it.

3.2.74 **span:** A component in the summarize abstract-operation result containing the lowest and highest sequence-numbers of the entries that matched the selection criteria.

3.2.75 **stored-messages:** The most important information-base in this Recommendation, used to store entries containing messages and reports delivered by the MTS to the MS.

3.2.76 **subscription:** A long-term agreement between the MS supplier or administrator and the MS customers (MS-owners) on the availability and use of optional MS features such as optional services and attributes. This Recommendation, assumes that such a mechanism is provided, but does not prescribe or offer any standardized method for how to provide this.

3.2.77 **substring:** A filter-item used to specify string of characters which appear (in the same given order) in a value of an attribute.

3.2.78 **summarize abstract-operation:** An abstract-operation which allows a quick overview of the kind and number of entries which are currently stored in an information-base.

3.2.79 **synopsis:** A content specific attribute that may be used to show how child-entries, containing parts of the content, are related to each other and the main-entry. The attribute has to be specified in the Recommendation, which describes the content-type, e.g. see IPM-synopsis defined in Recommendation X.420.

4 Abbreviations

For a list of abbreviations refer to Recommendation X.402.

5 Conventions

This Recommendation uses the description conventions listed in the following four paragraphs.

5.1 *Conventions for abstract-services*

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

1) ASN.1 itself, to specify the abstract-syntax of information-bases and their components, and common data-types.
2) The ASN.1 PORT macro and associated abstract-service definition conventions of Recommendation X.407, to specify the retrieval port.
3) The ASN.1 ABSTRACT-BIND, ABSTRACT-UNBIND, ABSTRACT-OPERATION, and ABSTRACT-ERROR macros and associated abstract-service definition conventions of Recommendation X.407, to specify the MS abstract-service.

Whenever this Recommendation describes a class data structure having components, each component is categorized as one of the following **grade**:

1) **Mandatory** (M) – A mandatory component shall be present in every instance of the class.
2) **Optional** (O) – An optional component shall be present in an instance of the class at the discretion of the object (e.g. user) supplying that instance.
3) **Conditional** (C) – A conditional component shall be present in an instance of the class as dictated by this Recommendation.

5.2 *Conventions for attribute-types used in Table 1/X.413 (§ 11)*

This Recommendation uses the conventions listed below in its definition of the attribute-types for the 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 *Support level by the MS and access UA* the following values can occur:

M mandatory

O optional

For the columns headed *Presence in delivered message entry*, *Presence in delivered report entry*, and *Presence in returned message entry*, the presence of each attribute-type is described by one of the following values:

P *always present* in the entry because:
- it is mandatory for generation by the MS; or
- it is a mandatory or defaulted parameter in the relevant abstract-operation.

C *conditionally present* in the entry. It would be present because:
- it is supported by the MS and subscribed to by the user and;
- it was present in an optional parameter in the relevant abstract-operation.

– *always absent*, otherwise.

For the columns headed *Available for list, alert* and *available for summarize,* the following values can occur:

N no

Y yes

5.3 *Conventions for attribute-types used in Table 2/X.413 of (§ 11)*

This Recommendation uses the conventions listed below in its definition of the attribute-type for the MS abstract-service. Paragraph 11 includes Table 2/X.413 that lists the attribute-types.

For the column headed *single/multi-valued* the following values can occur:

S single-valued

M multi-valued

For the column headed *Source generated by* the following values can occur:

MD MessageDelivery abstract-operation

MS MessageStore

RD ReportDelivery abstract-operation

5.4 *Font conventions for text in general*

Throughout this Recommendation, terms are rendered in **bold** when defined, without emphasis upon all other occasions. Terms that are proper nouns are capitalized, generic terms are not. Multi-word generic terms are hyphenated.

5.5 *Font conventions for ASN.1 definitions*

Throughout this Recommendation, ASN.1 definitions are written in a different (**bold**) font than the rest of the document in order to highlight the difference between normal text and ASN.1 definitions. The font used for ASN.1 definitions is also one size smaller than the ordinary text. When ASN.1 protocol elements and elements values are described in accompanying text, their names are rendered in **bold**.

5.6 *Rules for ASN.1 definitions*

ASN.1 definitions appears both in the body of the document to aid the exposition, and again, formally in Annexes for reference. If differences are found between the ASN.1 used in the exposition and that formally defined in the correspoding Annex, a specified error is indicated.

6 Message store model

The message store (MS) is modeled as an atomic object, wich acts as a provider of services to an MS abstract-service-user (i.e., a user agent), and a user of the services provided by the message transfer system (MTS).

The MS serves an intermediary role between the UA and the MTS. Its primary function is to accept delivery of messages on behalf of a single MHS end-user, and to retain them for subsequent retrieval by the end-user's UA. The MS also provides indirect message-submission and message-administration services to the UA, in effect, via "pass-through" to the MTS. This enables the MS to provide additional functionality compared to submission directly to the MTA; such a forwarding of messages residing in the MS.

Like the UA, the MS acts on behalf of only a single MHS end-user; i.e. it does not provide common or shared multi-user MS service.

The MS is described using an abstract model in order to define the sevices provided by the MS – the Message Store abstract-service. Figure 1/X.413 shows the MS abstract-service in relation to its user and to the Message Transfer System abstract-service. In this figure, the open boxes represent the consumption of the abstract service, and the closed boxes represent the supply of the abstract service.

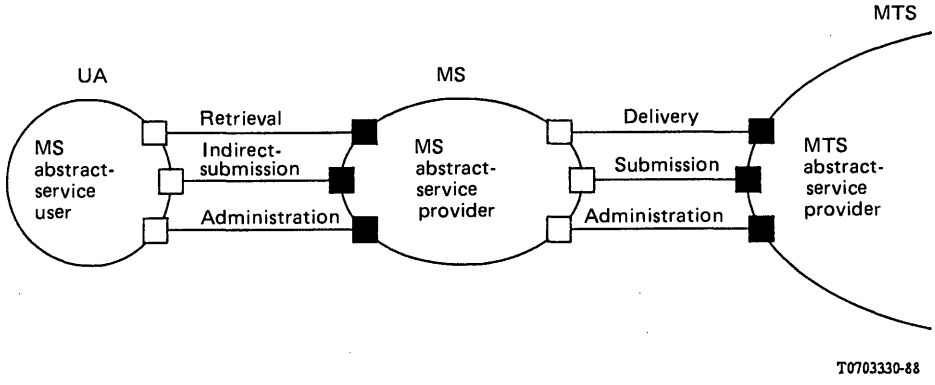


FIGURE 1/X.413

Message store abstract-service

For an introduction and description of the abstract-service concept and its definition conventions, see Recommendation X.407.

In secure messaging the MS is treated as a separate object with a unique identity and has separate key (or a set of keys) to the UA.

6.1 *Message store object*

The MS is modeled as an atomic object. It supplies the MS Retrieval Port abstract-services to the MS abstract-service-user. Acting as a "surrogate" MTS abstract-service-provider, the MS also supplies the MTS submission and administration abstract-service to the MS abstract-service-user (MS-user), and acting as a UA "surrogate", it consumes the MTS delivery port, submission port, and administration port abstract-services in its role as MTS abstract-service user.

The formal definition for the message store object is as follows:

```
mS OBJECT
        PORTS { retrieval[S],
                indirectSubmission[S],
                administration[S],
                delivery[C],
                submission[C],
                administration[C]}
        ::= id-ot-ms
```

The MS-user is also modeled as an object. It consumes the MS Retrieval Port and Indirect-submission Port abstract-services and the Administration Port abstract-services provided transparently by the MS.

```
msUser OBJECT
        PORTS { retrieval[C],
                indirectSubmission[C],
                administration[C]}
        ::= id-ot-ms-user
```

6.2 *Message store ports*

An MS provides the retrieval, indirect-submission, and administration ports to the MS abstract-service user. The collection of capabilities provided by these port provides the MS abstract-service. The retrieval capabilities are unique to the MS. These capabilities include obtaining information on, fetching (in whole or in part), and deleting messages residing in the MS. Additional capabilities are provided for registering certain MS provided automatic actions (i.e., auto-forwarding and alert).

Note – ISO are planning to define additional message management services performed by the MS on the UA's behalf, for logging incoming and outgoing messages, and for auto-correlating incoming notifications with logging information about outgoing messages. These are outside the scope of this CCITT Recommendation.

In order to provide the services described in § 6.1 to an MS-user, the MS interacts, on behalf of the MS-user, with the MTS abstract-service, and acts as a consumer of the MTS delivery, submission and administration ports. The abstract-services provided by the MTS ports are defined in clause 8 of Recommendation X.411.

By means of the abstract-bind operation, the MS authenticates an MS-user before providing it with any of the above retrieval capabilities. Similarly, the MTS abstract-services must authenticate the MTS abstract-service user before extending its services to the MTS abstract-service-user.

With the exception of the retrieval port provided alert service and the indirect-submission port provided submission-control service, all the services provided by the MS abstract-service are invoked by the MS-user and performed by the MS.

Security-labels may be assigned to the MS in line with the security-policy in force. The security-policy may also define how security-labels are to be used to enforce the security-policy. If security-labels are assigned to the MS, the handling of stored messages and reports are not assigned to the MS, the handling of stored-messages and reports is discretionary.

If security-contexts are established between the UA and the MS, and between the MS and the MTA, the security-label that is assigned to a message or probe is confined by the security-context in line with the security-policy in force. If security-contexts are not established the assignment of a message-security-label to a message or probe is at the discretion of the originator.

6.2.1 *Retrieval port*

The **retrieval port** is defined as follows:

```
retrievalPORT
        CONSUMER INVOKES{
            Summarize,
            List,
            Fetch,
            Delete,
            Register-MS}
        SUPPLIER INVOKES{
            Alert}
        ::= id-pt-retrieval
```

The details of the **retrieval port** abstract-services are described in §§ 7 to 9.

6.2.2 *Indirect-submission port*

The **indirect-submission port** is defined as follows:

```
indirectSubmissionPORT ::= submission
```

The **indirect-submission port** makes use of the submission port abstract-services defined in § 8.2 of Recommendation X.411.

6.2.3 *Administration port*

The **administration port** is defined in § 8.4 of Recommendation X.411.

The MS shall have no interaction with the change-credentials abstract-service. If the MS-user needs to have its credentials updated, then the register-MS abstract-operation is used. See § 8.6.

6.3 *Information model*

This paragraph describes the information model used by the MS. It models **information-bases**, which consist of **entries**, which consist of **attributes**.

6.3.1 *Information-bases*

The MS stores and maintains **information-bases** in the MS is a "data-base" containing all the **entries** representing constituent objects of a particular category or categories.

This Recommendation defines and describes the **stored-messages information-base**. This holds information derived from message-deliveries and report-deliveries to the MS across the MTS Delivery Port, and is described in § 6.4.

Note – A future Addendum to the corresponding Part of the ISO Standard will define additional information-bases for logging, called the inlog and outlog, which are outside the scope of this CCITT Recommendation.

```
informationBase ::= INTEGER{
        stored-messages     (0),
        inlog               (1),
        outlog              (2) } (0 .. ub-information-bases)
```

6.3.2 *Entries*

Each **information-base** is organized as a sequence of **entries**. An **entry** represents a single object (such as a delivered message) within the **information-base**.

Each entry is identified by means of its **sequence-number**, unique within an **information-base**, and generated by the MS as new entries are created. Within an **information-base**, the MS generates the **sequence-numbers** in ascending order without cycling, and they are never re-used.

```
SequenceNumber ::= INTEGER (0 .. ub-messages)
```

Note – For example, the MS may choose to allocate **sequence-numbers** by using the time to a sufficient granularity to ensure uniqueness.

6.3.3 *Attributes*

6.3.3.1 *Introduction*

An **entry** consists of a set of **attributes**. This is depicted in Figure 2/X.413.

Each **attribute** provides a piece of information about, or derived from, the data to which the **entry** corresponds. One such piece of information is the **sequence-number** of the **entry** itself, and another is the **creation-time**.

An **attribute** consists of an **attribute-type**, which identifies the class of information given by an **attribute**, and the corresponding **attribute-value(s)**, which are particular instances of that class appearing in the **entry**.

```
Attribute ::= SEQUENCE{
        type        AttributeType,
        values      SEQUENCE SIZE (1 .. ub-attribute-values) OF ANY --DEFINED BY type--}
```

Note – Thus, for example, in a delivered-message-entry (described in § 6.4) the **attribute-type** could be the message's **priority**, and a corresponding **attribute-value** could be **urgent**.

All **attributes** in an **entry** must be of distinct **attribute-types**.

For some **attribute-types**, an **attribute** may only contain a single **attribute-value**. Such an **attribute-type** is said to be **single-valued**. For others, an **attribute** may contain one or more **attribute-values**, all of the same ASN.1 data-type. Such an **attribute-type** is said to be **multi-valued**. Whether an **attribute-type** is **single-valued** or **multi-valued** is stated when the **attribute-type** is defined (see § 6.3.3.2).

Note 2 – Thus, for example, the **attribute-type** for the **originator-name attribute** (described in § 11.2.28) is **single-valued**, whereas that for **other-recipient-names** (described in § 11.2.29) is **multi-valued**.

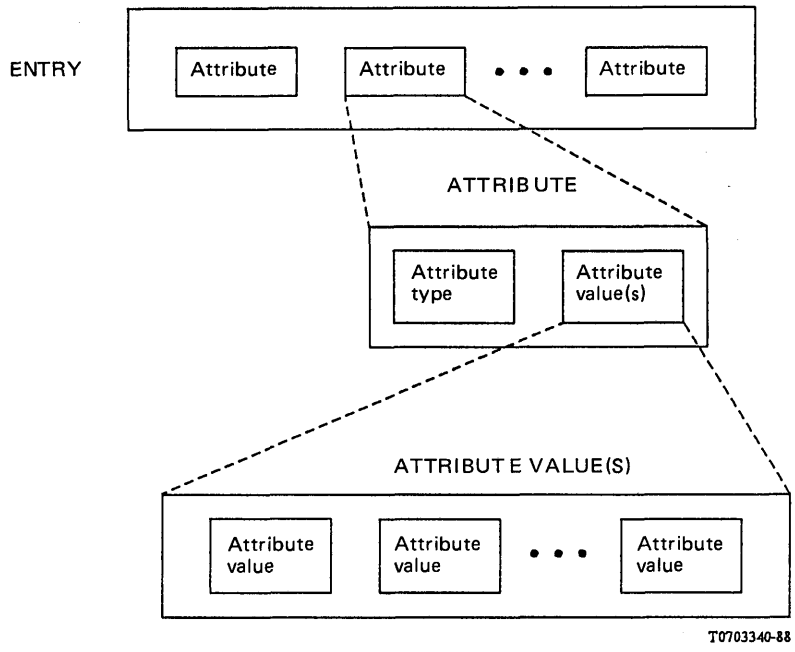


FIGURE 2/X.413

The components of an entry

6.3.3.2 *Attribute-type*

Some **attribute-types** will be internationally standardized. Other **attribute-types** will be defined by national administrative authorities and private organizations. This implies that a number of separate authorities will be responsible for assigning types in a way that ensures that each is distinct from all other assigned types. This is accomplished by identifying each **attribute-type** with an object-identifier when the **attribute-type** is defined.

AttributeType ::= OBJECT IDENTIFIER

Certain general-purpose **attribute-types** for the stored-messages information-base are defined in § 11. Such **attribute-types** are known as **general-attribute-types** and attributes of these types as **general-attributes**.

6.3.3.3 *Attribute-values*

Defining an **attribute-type** also involves specifying the ASN.1 data-type to which every value in such attributes must conform. The data-type of an **attribute-value** for the **attribute-type** is defined through the object-identifier for the **attribute-type**.

6.3.3.4 *Attribute-type definition and the ATTRIBUTE macro*

The definition of an **attribute-type** involves:

a) assigning an object-identifier to the **attribute-type**;

b) indicating the ASN.1 data-type of an **attribute-value**;

c) indicating whether an **attribute** of this **attribute-type** may have more than one value;

d) indicating whether an **attribute** of this **attribute-type** may be used for filtering based on equality, substrings, and/or ordering relations (see § 8.1.2).

Note – A filter may always test for the presence or absence in an entry of an **attribute** of a particular **attribute-type**.

The following ASN.1 macro is used to define an **attribute-type**. The formal definition of this macro is given in Recommendation X.501 and is documented here as an aid to the reader.

```
ATTRIBUTE MACRO ::=
BEGIN

    TYPE NOTATION    ::= AttributeSyntax Multivalued | empty
    VALUE NOTATION   ::= value (VALUE OBJECT IDENTIFIER)

    AttributeSyntax  ::= "WITH ATTRIBUTE-SYNTAX" SyntaxChoice
    SyntaxChoice     ::= value (ATTRIBUTE-SYNTAX) Constraint | type MatchTypes

    Constraint       ::= "(" ConstraintAlternative ")" | empty
    ConstraintAlternative ::= StringConstraint | IntegerConstraint
    StringConstraint ::= "SIZE" "(" SizeConstraint ")" | empty
    SizeConstraint   ::= SingleValue | Range
    SingleValue      ::= value (INTEGER)
    Range            ::= value (INTEGER) ".." value (INTEGER)
    IntegerConstraint ::= "(" Range ")"

    MatchTypes       ::= "MATCHES FOR" Matches | empty
    Matches          ::= Match Matches | Match
    Match            ::= "EQUALITY" | "SUBSTRINGS" | "ORDERING"
    Multivalued      ::= "SINGLE VALUE" | "MULTIVALUE" | empty

END
```

The correspondence between the parts of the definition, as listed above, and the various pieces of the notation introduced by the **ATTRIBUTE** macro, is as follows:

a) **MACRO value**: The **object-identifier** which is used to identify an attribute.

b) **Attribute-syntax**: Notes which syntax-choice has been made.

c) **Syntax-choice**: Notes whether the attribute is defined externally or internally. The syntax of all the attributes defined in this [Recommendation Part of the Standard] is defined internally, which means using the choice **typeMatchTypes**.

d) **Multivalued**: denotes whether the attribute is single or multi-valued.

e) **Match-types**: Gives the data-type of the contents of the attribute, and describes whether the **attributes** can be matched **("MATCHES FOR")** for **equality ("EQUALITY")**, for **substrings ("SUBSTRINGS")**, and for an **ordering** relation **("ORDERING")**. If the production is empty, then no matching rules are defined.

Matching for this Recommendation is restricted as follows:

i) **EQUALITY** is applicable to any attribute-syntax. The presented value must conform to the data-type of the attribute-syntax;

ii) **SUBSTRING** is applicable to any attribute-syntax with a **string** data type. The presented value must be a sequence ("SEQUENCE OF"), each of whose elements conforms to the data-type, and

iii) **ORDERING** is applicable to any attribute-sytanx for which a rule can be defined that will allow a presented value to be described as less than equal to, or greater than a target value. The presented value must conform to the data-type of the attribute-syntax. MS uses this for the INTEGER and UTCTime data types. For UTCTime, the ordering is chronological, not alphabetical.

The remaining choices and parameters of the **ATTRIBUTE** macro are not used in this Recommendation.

6.3.4 *Main-entries, parent-entries, and child-entries*

Although entries in a single information-base are generally independent of each other, the MS information model allows such **entries** to be related to one another. One entry, a **child-entry**, may be the child of another, its **parent-entry**, in a tree-structured relationship. An entry which is not a **child-entry** is termed a **main-entry**.

This relationship is recorded by means of two special **general-attributes**:

a) **parent-sequence-number**: This single-valued attribute gives the sequence-number of a **child-entry**'s **parent-entry**. It is absent from a **main-entry**. Its definition is given in § 11.2.30.

b) **child-sequence-numbers**: This multi-valued attribute gives the sequence-numbers of all the **child-entries** of a **parent-entry**. It is absent from an entry which is not a **parent-entry**. Its definition is given in § 11.2.1.

The abstract-operations of the MS abstract-service (see § 8) act by default only on **main-entries**. Some may be directed to act on all entries, both **main-entries** and **child-entries**. In particular, the argument of a delete abstract-operation (see § 8.5) may only select **main-entries**, in which case the **main-entry** and all its children and children's children, etc., will also be deleted.

Note – This concept allows, for example, those body-parts of an interpersonal message which contain a forwarded message (for details see § 19.1 of Recommendation X.420) to be represented by individual **child-entries**. The **content general-attribute** of the **main-entry** will comprise the complete **content**, so the data representing that message **body-part** is logically present in more than one **entry**.

6.4 *Stored-messages*

The **stored-messages information-base** acts as a repository for information obtained from the Message Delivery and ReportDelivery abstract-operation of the DeliveryPort. It contains entries for delivered messages (**delivered-message-entries**), of an open-ended number of content-types, and for reports (**delivered-report-entries**). An entry in the **stored-message-information-base** is created by the MS when a message or report is delivered to the MS. For more details of these entries and how they are generated, see §§ 11 and 15.

To draw information from the content of a message, the MS must know the content's syntax and semantics, as signaled via the content-type. In general, a particular instance of the MS has knowledge of zero or more content-types. When an MS encounters a message of whose content-type it has insufficient knowledge, it is unable to generate any content-type-specific attributes in the message's entry.

A delivered-message or an arriving notification may result in a main-entry and one or more levels of child-entries. The one case defined by this Recommendation is when a non-delivery notification contains a returned-content (the **delivered-report-entry** is the main-entry and the returned-content is its child-entry, known as a **returned-content entry**).

The rules for how a message-content may be split across several entries is specific to each content-type. A content-specific **synopsis-attribute** may be used to show how the main-entry and the corresponding child-entries are related. When such an attribute is defined, it appears in the Recommendation which defines the content-type itself. The **synopsis-attribute** is constructed by the MS.

Note – For Interpersonal Messaging (Recommendation X.420), nested IP-messages within an IP-message are each represented by a child-entry. The ipm-synopsis attribute-type is an example of a content-specific **synopsis-attribute-type**.

An important property of an entry in the stored-messages is its **entry-status**. It is created and maintained by the MS. It can take the following values:

a) **New** – The message has neither been **listed** by a UA nor has it been automatically processed by the MS.

b) **Listed** – Information about the message has been returned to the UA in either a list abstract-operation or a fetch abstract-operation, but the message has not yet been completely **processed**.

c) **Processed** – Either a UA has "completely fetched" the message, or the MS has performed some auto-action on it. (Note that some auto-actions result in the message being deleted). The exact definition of "completely fetched" is content-specific and is defined in the corresponding content-specific Recommendation.

   The **entry-status** of a (non-)delivery-notification becomes **processed** when the delivered-report-envelope is retrieved.

The definition for **entry-status** is as follows:

```
EntryStatus ::= INTEGER {
        new             (0),
        listed          (1),
        processed       (2)}
```

6.5 *Auto-actions*

6.5.1 *Introduction*

This paragraph defines a framework for automatic actions (**auto-actions**) which may be registered with the MS.

An **auto-action** is an action that will occur automatically whenever the associated registration criteria have been satisfied. The result of an action being invoked is visible externally to the MS. **Auto-actions** are registered in the MS using the Register-MS abstract-operation (see § 8.6).

Each class of **auto-action** is identified by means of an **auto-action-type**. Associated with the registration of an **auto-action**, there is a corresponding **auto-action-registration-parameter**, which are the parameters needed by the MS to perform the registered **auto-action** automatically. The registration of an **auto-action** requires the use of an **auto-action-registration-identifier** to identify the particular registration.

```
AutoActionRegistration ::= SEQUENCE {
        type                        AutoActionType,
        registration-identifier     [0] INTEGER (1 .. ub-per-auto-action)DEFAULT1,
        registration-parameter      [1] ANY DEFINED BY type }
```

6.5.2 *Auto-action-type*

Some **auto-action-types** will be internationally standardized. Other **auto-action-types** will be defined by national administrative authorities and private organizations. This implies that a number of separate authorities will be responsible for assigning types in a way that ensures that each is distinct from all other assigned **auto-action-types**. This is accomplished by identifying each **auto-action-type** with an object identifier when the **auto-action-type** is defined.

```
AutoActionType ::= OBJECT IDENTIFIER
```

Certain general-purpose **auto-action-types** are defined in § 12. Such **auto-action-types** are known as **general-auto-action-types** and **auto-actions** of these types as **general-auto-actions**.

6.5.3 *Auto-action-registration-parameter*

Defining an **auto-action-type** also involves specifying the ASN.1 data-type to which the **auto-action-registration-parameter** must conform. The data-type of an **registration-parameter** is defined through the object-identifier for the **auto-action-type**.

6.5.4 *Auto-action-type definition and the AUTO-ACTION macro*

The definition of an **auto-action-type** involves:

a) assigning an object-identifier to the **auto-action-type**;

b) indicating the ASN.1 data-type of the **auto-action-registration-parameter**.

The following ASN.1 macro may (but need not) be used to define an **auto-action-type**:

```
AUTO-ACTION MACRO ::=
BEGIN
        TYPE NOTATION   ::= Registration
        VALUE NOTATION  ::= value (VALUE OBJECT IDENTIFIER)

        Registration    ::= "REGISTRATION PARAMETER IS" type
END
```

The correspondence between the parts of the definition, as listed above, and the various pieces of the notation introduced by the **AUTO-ACTION** macro, is as follows:

a) **Registration**: gives the data-type of the registration parameters association with an auto-action.

b) **Value**: the **object-identifier** which is used to identify the auto-action.

Note – No support is provided in the macro for defining the interaction (if any) between different registrations of the same (or different) **auto-actions**.

6.6 *Forwarding of messages*

The MS-user makes use of the message-submission abstract-operation and its parameters as defined in § 8.2 of Recommendation X.411 to request that a message stored in the MS be explicitly forwarded to other users.

The **forwarding-request parameter** is defined using the **EXTENSION** macro defined in § 9 of Recommendation X.411 as follows:

```
forwarding-request EXTENSION
        SequenceNumber
        CRITICAL FOR SUBMISSION
        ::= 36
```

If the **sequence-number** supplies does not match that of an entry into the **stored messages information-base**, or matches an entry that is unsuitable for forwarding, this is reported using the **inconsistent-request** abstract-error of § 8.2.2.7 of Recommendation X.411.

7 Abstract-bind and abstract-unbind operations

7.1 *Abstract-bind-operation*

The **MS-bind abstract-bind-operation** binds the indirect-submission, retrieval and administration ports of the MS-user (consumer) to the MS (supplier). The initiator (of the MS-bind) is the MS-user, while the responder is the MS itself. MS-bind is defined as follows:

```
MSBind ::= ABSTRACT-BIND
        TO { IndirectSubmission[5], retrieval[5], administration[5] }
        BIND
            ARGUMENT    MSBindArgument
            RESULT      MSBindResult
            BIND-ERROR  MSBindError
```

Only one abstract-association may exist at any one time between the MS and the MS-user.

7.1.1 *Abstract-bind-argument*

The **abstract-bind-argument** parameters are used to identify, authenticate and set the security-context for an MS abstract-service-user. They also contain a set of restrictions for entries to be returned as result of a Fetch abstract-operation, and finally, a request to be informed of the auto-action-types, attribute-types and content-types supported by the MS.

The definition of these parameters is as follows:

```
MSBindArgument ::= SET{
        initiator-name          ORAddressAndOrDirectoryName,
        initiator-credentials   [2] InitiatorCredentials,
        security-context        [3] IMPLICIT SecurityContext OPTIONAL,
        fetch-restrictions      [4] Restrictions OPTIONAL -- default is none --,
        ms-configuration-request [5] BOOLEAN DEFAULT FALSE }
```

1) **Initiator-name** (C): This argument contains the name of the initiator of the association and is supplied by the initiator. This argument is defined further in § 8.1.1.1.1.1 of Recommendation X.411.

2) **Initiator-credentials** (M): This parameter contains the **credentials** of the initiator of the association. It shall be generated by the initiator of the abstract-association.

   The **initiator-credentials** may be used by the responder to authenticate the identity of the initiator (see Recommendation X.509).

   If only **simple-authentication** is used, the **initiator-credentials** comprise a simple password.

   If **strong-authentication** is used, the **initiator-credentials** comprise an **initiator-bind-token**, and, optionally, an **initiator-certificate**. The **initiator-bind-token** and **initiator-certificate** are defined further in § 8.1.1.1.1.2 of Recommendation X.411. The **initiator-credentials** of the MS-user may differ from the **initiator-credentials** used in the **MTS-bind** as defined in § 8.1.1.1.1.2 of Recommendation X.411.

3) **Security-context** (O): This parameter identifies the **security-context** that the initiator of the abstract-association proposes to operate at. It is generated by the initiator of the abstract-association. The **security-context** is defined further in § 8.1.1.1.1.3 of Recommendation X.411.

The **security-context** comprises one or more **security-labels** that define the sensitivity of interactions that may occur between the MS abstract-service-user and the MS-abstract-service for the duration of the abstrct-association, in line with the **security-policy** in force. The **security-context** shall be one that is allowed by the registered **user-security-labels** of the MS-abstract-service-user and by the **security-labels** with the MS.

In the absence of this parameter, **security-contexts** are not established between the MS-abstract-service-user and the MS-abstract-service is at the discretion of the invoker of the abstract-service.

4) **Fetch-restrictions** (O): This contains the restrictions on entries to be returned as result of a fetch abstract-operation. The **fetch-restrictions** remain set until an abstract-unbind-operation is issued.

In the absence of this argument, the default is that no **fetch-restrictions** need to be performed.

This argument consists of the following components:

```
Restrictions ::= SET{
    allowed-content-types       [0] SET SIZE (1 .. ub-content-types) OF OBJECT IDENTIFIER
                                    OPTIONAL
        -- default is no restriction --,
    allowed-EITs                [1] MS-EITs OPTIONAL -- default is no restriction --
    maximum-content-length      [2] ContentLength OPTIONAL -- default is no restriction --
```

a) **Allowed-content-types** (C): The content-types that the MS abstract-service-user is prepared to accept as result of a Fetch abstract-operation. Any message with a content-type other than the ones specified will not be returned, but result in an error, unless the Fetch abstract-operation has explicitly overriden the restriction.

In the absence of this component, the default is that no fetch-restrictions on content-types need to be performed.

b) **Allowed-EITs** (C): The encoded-information-types that the MS abstract-service-user is prepared to accept as result of a fetch abstract-operation. If a message contains encoded-information-types other than the ones specified, a filtering will take place so that disallowed EIT parts are not returned along with the text of the message. If the whole message consists of disallowed EITs, an error will be reported. No filtering will take place if the fetch abstract-operation has explicitly overridden the restriction.

```
MS-EITs ::= SET SIZE (1 .. ub-encoded-information-types) OF MS-EIT

MS-EIT ::= OBJECT IDENTIFIER
```

In the absence of this component, the default is that no **fetch-restrictions** on encoded-information-types need to be performed.

c) **Maximum-content-length** (C): The maximum content length that the MS-abstract-service-user is prepared to accept as a result of a fetch abstract-operation. Any message with a **content-length** exceeding the one specified will not be returned, but result in an error, unless the fetch abstract-operation has explicitly overridden the restriction.

In the absence of this component, the default is that no **fetch-restrictions** on **content-length** need to be performed.

5) **MS-configuration-request** (C): The **MS-configuration-request** is requested to obtain information relating to which auto-actions and optional attributes the MS provides support for.

In the absence of this component, the default is false which indicates that no such request is being made.

7.1.2 *Abstract-bind-result*

The abstract-bind-result parameters are as follows:

```
MSBindResult ::= SET{
        responder-credentials       [2] ResponderCredentials,
        available-auto-actions      [3] SET SIZE (1 .. ub-auto-actions) OF AutoActionType OPTIONAL,
        available-attribute-types   [4] SET SIZE (1 .. ub-attributes-supported) OF Attribute Type
                                        OPTIONAL,
        alert-indication            [5] BOOLEAN DEFAULT FALSE,
        content-types-supported     [6] SET SIZE (1 .. ub-content-types) OF OBJECT IDENTIFIER
                                        OPTIONAL }
```

1) **Responder-credentials** (M): This parameter contains the credentials of the responder of the abstract-association. It shall be generated by the responder of the abstract-association.

   The **responder-credentials** may be used by the initiator to authenticate the identity of the responder (see Recommendation X.509).

   If only **simple-authentication** is used, the **responder-credentials** comprise a simple **password** associated with the responder.

   If **strong-authentication** is used, the **responder-credentials** comprise a **responder-bind-token**, and, optionally, a **responder-certificate**, both of which are generated by the responder of the abstract-association. The **responder-bind-token** and **responder-certificate** are defined further in § 8.1.1.1.2.2 of Recommendation X.411.

2) **Available-auto-actions** (C): Specifies the set of all possible **auto-actions** that are supported by the MS (not just those requested by the MS abstract-service-user). Only present if an **MS-configuration-request** is made.

3) **Available-attribute-types** (C): Specifies the set of all optional attributes supported by the MS. Only present if an **MS-configuration-request** is made.

4) **Alert-indication** (C): If true then an **alert** condition has occurred since the last successful **Alert-indication.**

5) **Content-types-supported** (C): Specifies a set of object-identifiers that define the **content-types** that the MS has knowledge of. Only present if an **MS-configuration-request** is made.

7.1.3 *Abstract-bind-errors*

There are two possible errors defined by the **retrieval port**, namely **authentication-error** and **unacceptable-security-context.**

The definition of the errors is:

```
MSBindError ::= ENUMERATED{
        authentication-error                (0),
        unacceptable-security-context       (1),
        unable-to-establish-association     (2)}
```

1) **Authentication-error** (C): This error reports that an abstract-association cannot be established because the initiator's **credentials** are not acceptable or are improperly specified.

   The **authentication-error** has no parameters.

2) **Unacceptable-security-context** (C): This error reports that the **security-context** proposed by the initiator of the abstract-association is unacceptable to the responder.

   The **unacceptable-security-context** error has no parameters.

3) **Unable-to-establish-association** (C): This error reports that the responder has rejected the initiator's attempt to establish an abstract-association.

   The **unable-to-establish-association** error has no parameters.

7.2 *Abstract-unbind-operation*

The **MS-unbind abstract-unbind-operation** closes the abstract-association. The issuing of an **abstract-unbind-operation** results in the relaxation of any **fetch-restrictions** that were specified in the **abstract-bind operation** argument. There is no argument, result, or error associated with the **abstract-unbind-operation.**

```
MSUnbind ::= ABSTRACT-UNBIND
        FROM { indirectSubmission[S], retrieval[S], administration[S] }
```

8 Abstract-operations

This paragraph defines the following **abstract-operations** available at the retrieval port:

a) summarize;

b) list;

c) fetch;

d) delete;

e) register-MS;

f) alert.

The MS is the MS abstract-service-provider of each of these **abstract-operations**. For the formal definition of the retrieval port, see § 6.2.

The abstract-operations may be performed asynchronously subject to the following conditions. The delete and register-MS abstract-operations shall not be performed until all outstanding abstract-operations have been completed. Additionally these abstract-operations are performed in the order in which they are invoked and are required to complete prior to any other abstract-operations being performed. As a consequence of this and the fact that the list and fetch abstract-operations change the status of a message entry, the results of the summarize, list and fetch abstract-operations may be non-deterministic.

8.1 *Common-data-types used in abstract-operations*

This paragraph defines a number of common data-types which are used in several of the **abstract-operations** defined in the remainder of § 8. Many of the **abstract-operations** also make use of entries and attributes as defined in § 6.3.

The common data-types defined in this Recommendation are:

a) range;

b) filter;

c) selector;

d) entry information selection;

e) entry information.

8.1.1 *Range*

A **range** parameter is used to select a contiguous sequence of entries from an information-base.

```
Range ::= CHOICE {
        sequence-number-range    [0] NumberRange,
        creation-time-range      [1] TimeRange }

NumberRange ::= SEQUENCE {
        from    [0] SequenceNumber OPTIONAL -- omitted means no lower bound --,
        to      [1] SequenceNumber OPTIONAL -- omitted means no upper bound -- }

TimeRange ::= SEQUENCE {
        from    [0] CreationTime OPTIONAL -- omitted means no lower bound --,
        to      [1] CreationTime OPTIONAL -- omitted means no upper bound -- }

CreationTime ::= UTCTime
```

The components of **range** have the following meanings:

1) **Sequence-number-range** (C), and

2) **Creation-time-range** (C): Both of these parameters identify the contiguous sequence of entries to be selected. The **sequence-number-range** is given in terms of **sequence-numbers**, and the **creation-time-range** is given in terms of **creation-times**. The **creation-time** of an entry is the time at which the MS generated the entry. The sequence numbers of successive entries are always in ascending order, but several adjacent entries may have the same **creation time**. The parameters of both **number-range** and **time-range** have the following meanings:

   a) **From** (O): This is the lower bound for the **range**.

      In the absence of this component, the default is **no lower bound**, and the selection starts with the earliest message (lowest **sequence-number**) in the information-base.

   b) **To** (O): This is the upper bound for the **range**.

      In the absence of this component, the default is **no upper bound**, and the selection finishes with the latest message (highest **sequence-number**) in the information-base.

8.1.2 *Filters*

8.1.2.1 *Filter*

A **filter** parameter applies a test to a particular entry and is either satisfied or not by the entry. The **filter** is expressed in terms of assertions about the presence or value of certain attributes of the entry, and is satisfied if and only if it evaluates to **true**.

```
Filter ::= CHOICE {
        item    [0] FilterItem,
        and     [1] SET SIZE (1 .. ub-nested-filters) OF Filter,
        or      [2] SET SIZE (1 .. ub-nested-filters) OF Filter,
        not     [3] Filter }
```

A **filter** is either a **filter-item**, or an expression involving simpler **filters** composed together using the logical operators **and**, **or**, and **not**.

Where the **filter** is:

a) an **item**, it is **true** if and only if the corresponding **filter-item** is **true**;

b) an **and**, it is **true** unless any of the **filters** in the **SET** are **false**.

Note – Thus, if there are no **filters** in the **SET**, the **and** evaluates to **true**.

c) an **or**, it is **false** unless any of the **filters** in the **SET** are **true**;

Note – Thus, if there are no **filters** in the **SET**, the **or** evaluates to **false**.

d) a **not**, it is **true** if and only if the **filter** is **false**.

8.1.2.2 *Filter-item*

A **filter-item** is an assertion about the presence or value(s) of an attribute of a particular type in the entry under test. Each such assertion is either **true** or **false**.

```
FilterItem ::= CHOICE {
        equality                [0] AttributeValueAssertion,
        substrings              [1] SEQUENCE {
            type                AttributeType,
            strings             SEQUENCE SIZE (1 .. ub-attribute-values) OF CHOICE {
              initital              [0] ANY -- DEFINED BY type --,
              any                   [1] ANY -- DEFINED BY type --,
              final                 [2] ANY -- DEFINED BY type -- } },
        greater-or-equal        [2] AttributeValueAssertion,
        less-or-equal           [3] AttributeValueAssertion,
        present                 [4] AttributeType,
        approximate-match       [5] AttributeValueAssertion }
```

Every filter-item includes an attribute-type which identifies the particular attribute concerned.

Any assertion about the value of such an attribute is only evaluated if the attribute-type is defined, and the purposed attribute-value(s) are of the data-type defined for attribute-values of that attribute.

Assertions about the value of an attribute by matching the attribute for EQUALITY, SUBSTRINGS, and ORDERING, as defined in § 6.3.3.4.

Where the **filter-item** asserts:

a) **equality**, it is **true** if and only if there is a value of the attribute which is equal to that asserted;

b) **substrings**, it is **true** if and only if there is a value of the attribute in which the specified **substrings** appear in the given order. The **substrings** must be non-overlapping, and may (but need not) be separated from the ends of the attribute-value and from one another by zero or more **string** elements.

The first character in **initial**, if present, shall match the first character in the attribute-value; the last character in **final**, if present, shall match the last character in the attribute-value. **any**, if present, may match any substring in the attribute-value;

c) **greater-or-equal**, it is **true** if and only if the relative ordering places the supplied value *after* any value of the attribute;

d) **less-or-equal**, it is **true** if and only if the relative ordering places the supplied value *before* any value of the attribute;

e) **present**, it is **true** if and only if such an attribute is present in the entry;

f) **approximate-match**, it is **true** if and only if there is a value of the attribute which matches that which is asserted by some locally-defined approximate matching algorithm (e.g. spelling variations, phonetic match, etc.) There are no specific guidelines for approximate matching in this version of the Recommendation. If approximate matching is not supported, this **FilterItem** should be treated as match for **equality**.

Note – If no matching rules are given in the attribute definition, this means that only the presence of the attribute can be tested in a **filter-item**.

8.1.2.3 *Attribute-value-assertion*

An **attribute-value-assertion** is a proposition, which may be **true**, **false**, or **undefined**, concerning the values of an entry. It involves an attribute-type and an attribute-value:

```
AttributeValueAssertion ::= SEQUENCE {
        type    AttributeType,
        value   ANY DEFINED BY type }
```

and is:

a) **underfined**, if any of the following holds:

1) the attribute-type is not present in the entry;

2) the definition of the attribute-type cannot be matched for equality or ordering;

3) the attribute-value does not conform to the data type of the attribute-values;

b) **true**, if the entry contains an attribute of that attribute-type, one of whose attribute-values matches that attribute-value;

c) **false**, otherwise.

8.1.3 *Selector*

A **selector** parameter is used to select entries from an information-base. The selection operates in three stages. Firstly, the total set of entries in the information-base may be restricted to particular contiguous set by specifying its range. Secondly, entries from within this set may be selected by specifying a filter which the selected entry must satisfy. Thirdly, a limit may be placed on the number of entries thus selected; in this case, it is those entries with the lowest sequence-numbers which are selected.

```
Selector ::= SET {
        child-entries   [0] BOOLEAN DEFAULT FALSE,
        range           [1] Range OPTIONAL -- default is unbounded --,
        filter          [2] Filter OPTIONAL -- default is all entries within the specified range --,
        limit           [3] INTEGER (1 .. ub-messages) OPTIONAL,
        override        [4] OverrideRestrictions OPTIONAL -- default is that any fetch-restrictions in force
                            do apply -- }
```

The components of **selector** have the following meanings:

1) **Child-entries** (O): If **false**, only main-entries are considered for selection. If **true**, both main-entries and child-entries are considered for selection.

   In the absence of this component, the default is *only main-entries are considered.*

2) **Range**(O): The abstract-syntax-notation of **range** is given in § 8.1.1.

   In the absence of this component, the default is *unbounded.*

3) **Filter** (O): The abstract-syntax-notation of **filter** is given in § 8.1.2.

   In the absence of this component, the default is *all entries within the specified range.*

4) **Limit** (O): This allows the specification of an upper limit on how many entries shall be selected.

   In the absence of this component, all of the selected entries will be returned.

   Note – The primary role of the limit is to protect against huge results from an abstract-operation as a consequence of badly formulated selections. It can also be used to give back an exact number of information-sets to fit a particular output-device.

5) **Override** (O): If an override of any of the **fetch-restrictions** is required, the corresponding component(s) of **override-restrictions** must be present.

```
OverrideRestrictions ::= BIT STRING {
    overrideContentTypesRestriction    (0),
    overrideEITsRestriction            (1),
    overrideContentLengthRestriction   (2) } (SIZE (1 .. ub-information-bases))
```

The bits of **override-restrictions** have the following meaning:

a) **Override-content-types-restriction** (M): This bit must be set to 1 if the **content-types-restriction** shall be overridden.

If this bit is set to 0, the **content-types-restrictions** as specified in the abstract-bind-operation will be applied.

b) **Override-EITs-restriction** (M): This bit must be set to 1 if the **EITs-restriction** shall be overridden.

If this bit is set to 0, the **EITs-restrictions** as specified in the abstract-bind-operation will be applied.

c) **Override-content-length-restriction**(M): This bit must be set to 1 if the **content-length-restriction** shall be overridden.

If this bit is set to 0, the **content-length-restrictions** as specified in the abstract-bind-operation will be applied.

In the absence of **override-restrictions**, the default is that all the **fetch-restrictions** as specified in the abstract-bind-operation will be applied.

8.1.4 *Entry-information-selection*

An **entry-information-selection** parameter indicates what information from an entry is being requested.

```
EntryInformationSelection ::= SET SIZE (0 .. ub-per-entry) OF AttributeSelection
```

An empty set indicates that information about the entry itself, rather than the attributes of entry, is being requested.

```
AttributeSelection ::= SET {
        type     AttributeType,
        from     [0] INTEGER (1 .. ub-attribute-values) OPTIONAL -- used if type is multi valued --,
        count    [1] INTEGER (1 .. ub-attribute-values) OPTIONAL -- used if type is multi valued -- }
```

The components of **attribute-selection** have the following meaning:

1) **Type** (M): This indicates the attribute-type of the attribute.

2) **From** (O): When an attribute is multi-valued, this integer gives the relative position of the first value to be returned. If it specifies a value beyond those present in the attribute, no values are returned. This component may only be present if the attribute-type is multi-valued. If it is omitted, values starting at the first value are returned.

3) **Count** (O): When an attribute is multi-valued, this integer gives the number of values to be returned. If there are less than **count** values present in the attribute, all values are returned. This component may only be present if the attribute-type is multi-valued. If it is omitted, there is no limit as to how many values are returned.

8.1.5 *Entry-information*

An **entry-information** parameter conveys selected information from an entry.

```
EntryInformation ::= SEQUENCE {
        sequence-number          SequenceNumber,
        attributes               SET SIZE (1 .. ub-per-entry) OF Attribute OPTIONAL }
```

The components of **entry-information** have the following meanings:

1) **Sequence-number** (M): The sequence-number identifying the entry. See § 6.3.2.2.

2) **Attributes** (O): The set of selected attributes from the entry. Where explicitly requested by a partial-attribute-request, a selected attribute that is defined to be multi-valued may contain a subset of all the attribute-values in the attribute as stored in the entry. This parameter is absent if information from the selected messages is not requested, for example, when the MS-abstract-service-user wants only the sequence-numbers of the selected messages.

8.2 *Summarize abstract-operation*

The **Summarize abstract-operation** returns summary counts of selected entries in an information-base. In addition to these summaries, a count of the entries selected, and their lowest and highest sequence-numbers are also returned. Zero or more individual summaries may be requested.

The **summarize abstract-operations** will only be successful when the information-base permits access according to the security-context and the enforced security-policy.

The attributes that may be used for summaries are restricted. For the general-attributes in the stored-messages information-base, the restrictions are given in Table 1/X.413.

```
Summarize ::= ABSTRACT-OPERATION
        ARGUMENT        SummarizeArgument
        RESULT          SummarizeResult
        ERRORS {
            AttributeError,
            InvalidParametersError,
            RangeError,
            SecurityError,
            SequenceNumberError,
            ServiceError }
```

Note – An example of the summarize abstract-operation is given in Annex F.

8.2.1 *Summarize-argument*

```
SummarizeArgument ::= SET {
        information-base-type [0] InformationBase DEFAULT stored-messages,
        selector              [1] Selector,
        summary-requests      [2] SEQUENCE SIZE (1 .. ub-summaries) OF AttributeType OPTIONAL
            -- absent if no summaries are requested -- }
```

The components of **summarize-argument** have the following meanings:

1) **Information-base-type** (O): This specifies which **information-base** is addressed by the abstract-operation. See § 6.3.1.

   In the absence of the **information-base-type** component, the default is stored-messages.

2) **Selector** (M): This is a set of selection criteria to determine which entries shall be summarized. See § 8.1.3.

3) **Summary-requests** (O): This is the sequence of attribute-types for which summaries are requested. This parameter is only present if a summary is requested.

8.2.2 *Summarize-result*

Should the request succeed, the **summarize-result** will be returned.

```
SummarizeResult ::= SET {
        next        [0] SequenceNumber OPTIONAL,
        count       [1] INTEGER (0 .. ub-messages) } -- of the entries selected --,
        span        [2] Span OPTIONAL -- of the entries selected, omitted if count is zero --,
        summaries   [3] Sequence SIZE (1 .. ub-summaries) OF Summary OPTIONAL)
```

The components of **summarize-result** have the following meanings:

1) **Next** (C): This is returned in the case where the number of entries selected would have been greater if it were not for the limit specified in the selector. The component contains the sequence-number for the next entry that would have been selected.

2) **Count** (M): This is an integer giving the count of entries that matched the selection criteria.

3) **Span** (C): This contains the lowest and highest sequence-numbers of the entries that matched the selection criteria. It is absent if there are no such entries.

```
Span ::= SEQUENCE {
    lowest   [0] SequenceNumber,
    highest  [1] SequenceNumber }
```

The components of **span** have the following meanings:

a) **Lowest** (M): This is the starting-point for the **span**, given as a sequence-number (see § 6.3.2.2).

b) **Highest** (M): This is the end-point for the **span** given as a sequence-number (see § 6.3.2.2).

4) **Summaries** (C): One **summary** is returned for each **summary-request**. The **summaries** are returned in the order that they were requested.

```
Summary ::= SET{
    absent    [0] INTEGER (1 .. ub-messages) OPTIONAL -- count of entries where the attribute is
                  absent --,
    present   [1] SET SIZE (1 .. ub-attribute-values) OF -- one for each attribute value present --
                      SEQUENCE{
                          type      AttributeType,
                          value     ANY DEFINED BY type,
                          count     INTEGER (1 .. ub-messages)} OPTIONAL}
```

The components of **summary** have the following meanings:

a) **Absent** (C): A count of the entries that do not contain an attribute of the attribute-type specified in the request. It is omitted if there are no such entries.

b) **Present** (C): A summary of the entries that contain an attribute of the attribute-type specified, broken down by the attribute-values actually present. It is omitted if there are no such entries.

The components of **present** have the following meanings:

i) **Type** (M): The type of the attribute.

ii) **Value** (M): The attribute-value for which the count is given.

iii) **Count** (M): A count of entries with this attribute-value.

8.2.3 *Summarize abstract-errors*

Should the request fail, one of the listed abstract-errors will be reported. The circumstances under which the particular abstract-errors will be reported are defined in § 9.

8.3 *List abstract-operation*

The **list-abstract-operation** is used to search a selected information-base for entries of interest and to return selected information from those entries.

The **list-abstract-operation** will only be successful when the information-base permits access according to the security-context and the enforced security policy.

The information that may be selected for entries in an information-base may be restricted. For the general-attributes in the stored-messages information-base, the restrictions are given in Table 1/X.413.

```
List ::= ABSTRACT-OPERATION
        ARGUMENT        ListArgument
        RESULT          ListResult
        ERRORS{
            AttributeError,
            InvalidParametersError,
            RangeError,
            SecurityError,
            SequenceNumberError,
            ServiceError}
```

8.3.1 *List-argument*

```
ListArgument ::= SET{
        Information-base-type     [0] InformationBase DEFAULT stored-messages,
        selector                  [1] Selector,
        requested-attributes      [3] EntryInformationSelection OPTIONAL}
```

The components of **list-argument** have the following meanings:

1) **Information-base-type** (O): This specifies which information-base is addressed by the abstract-operation. See § 6.3.1.

   In the absence of the **information-base-type** component, the default is stored-messages.

2) **Selector** (M): This is a set of selection criteria to determine which entries shall be returned. See § 8.1.3.

3) **Requested-attributes** (O): This indicates what information from the selected entries is to be returned in the result. See § 8.1.4.

   If this parameter is absent, the registered set of **list-attribute-defaults** is used. See § 8.6.1 for more information on these defaults.

8.3.2 *List-result*

Should the request succeed, the list-result will be returned.

```
ListResult ::= SET{
        next        [0] SequenceNumber OPTIONAL,
        requested   [1] SEQUENCE SIZE (1 .. ub-messages) OF EntryInformation OPTIONAL -- omitted
                        if none found -- }
```

The components of **list-result** have the following meanings:

1) **Next** (C): This is returned in the case where the number of entries selected would have been greater if it were not for the limit specified in the selector. The component contains the sequence-number for the next entry that would have been selected.

2) **Requested** (C): This conveys the requested entry-information (see § 8.1.5) from each selected entry (one or more), in ascending order of sequence-number. It is not present in the case that a search was performed and no entry was selected.

8.3.3 *List abstract-errors*

Should the request fail, one of the listed abstract-errors will be reported. The circumstances under which the particular abstract-errors will be reported are defined in § 9.

8.4 *Fetch abstract-operation*

The **fetch-abstract-operation** is used to return selected information from a specific entry in an information-base. Alternatively, it is used to return selected information from the first entry from among several entries of interest; in this case the sequence-numbers of the other selected entries are also returned. The **fetch-abstract-operation** will only be successful when information-bases permitted by the security-context and the security-policy in force are requested.

Information from an entry can be fetched several times, until the entry is explicitly deleted using the delete abstract-operation.

```
Fetch ::= ABSTRACT-OPERATION
        ARGUMENT        FetchArgument
        RESULT          FetchResult
        ERRORS{
            AttributeError,
            FetchRestrictionError,
            InvalidParametersError,
            RangeError,
            SecurityError,
            SequenceNumberError,
            ServiceError }
```

8.4.1 *Fetch-argument*

```
FetchArgument ::= SET{
        information-base-type   [0] InformationBase DEFAULT stored-messages,
        item                        CHOICE{
            search                      [1] Selector,
            precise                     [2] SequenceNumber },
        requested-attributes    [3] EntryInformationSelection OPTIONAL }
```

The components of **fetch-argument** have the following meanings:

1) **Information-base-type** (O): This specifies which information-base is addressed by the abstract-operation. See § 6.3.1.

   In the absence of the information-base-type component, the default is stored-messages.

2) **Item** (M): One of the components described below must be specified in order to determine which entry to fetch:

   a) **Search** (C): This is a selector specifying a set of entries of which the one with the lowest sequence-number is the entry to be fetched. See § 8.1.3.

   b) **Precise** (C): This is the sequence-number of the entry to be fetched. See § 6.3.2.2.

3) **Requested-attributes** (O): This indicates what information from the selected entry is to be returned in the result (see § 8.1.4).

   If this parameter is absent, the registered set of **fetch-attribute-defaults** is used. See § 8.6.1 for more information on these defaults.

8.4.2 *Fetch-result*

Should the request succeed, the **fetch-result** will be returned.

```
FetchResult ::= SET{
        entry-information    [0] EntryInformation OPTIONAL -- if an entry was selected --,
        list                 [1] SEQUENCE SIZE (1 .. ub-messages) OF SequenceNumber
                                 OPTIONAL,
        next                 [2] SequenceNumber OPTIONAL }
```

The components of **fetch-result** have the following meanings:

1) **Entry-information** (C): This is a set of attributes from one entry as requested in the argument. See § 8.1.5. It is not present in the case that a search was performed and no entry was selected.

2) **List** (C): This is returned in the case that a search was performed and more than one entry was found that matched the search selector. The list gives the sequence numbers, in ascending order, of these further entries.

3) **Next** (C): This is returned in the case where the number of entries selected would have been greater if it were not for the limit specified in the selector. The component contains the sequence-number for the next entry that would have been selected.

8.4.3 *Fetch abstract-errors*

Should the request fail, one of the listed abstract-errors will be reported. The circumstances under which the particular abstract-errors will be reported are defined in § 9.

8.5 *Delete abstract-operation*

The **delete abstract-operation** is used to delete selected entries from an information-base. A main-entry and all its dependent child-entries may only be deleted together. This is achieved by specifying just the main-entry as an argument. The delete abstract-operation will only be successful when operating on those information-bases permitted by the security-context and the security-policy in force.

For specific information-bases, there may be restrictions on which entries may be deleted. In addition, content specific actions may be taken as defined in the corresponding Recommendation which defines the content-type. For the stored-messages, no entry may be deleted if its entry-status (see § 6.4) is "new".

```
Delete ::= ABSTRACT-OPERATION
        ARGUMENT            DeleteArgument
        RESULT              DeleteResult
        ERRORS{
            DeleteError,
            InvalidParametersError,
            RangeError,
            SecurityError,
            SequenceNumberError,
            ServiceError }
```

8.5.1 *Delete-argument*

```
DeleteArgument ::= SET{
        information-base-type    [0] InformationBase DEFAULT stored-messages,
        items                        CHOICE{
            selector                     [1] Selector
            sequence-numbers             [2] SET SIZE (1 .. ub-messages) OF SequenceNumber}}
```

The components of **delete-argument** have the following meanings:

1) **Information-base-type** (O): This specifies which information-base is addressed by the abstract-operation. See § 6.3.1.

   In the absence of the **information-base-type** component, the default is stored-messages.

2) **Items** (M): One of the components described below must be specified in order to determine which entries to delete.

   a) **Selector** (C): See § 8.1.3.

   b) **Sequence-numbers** (C): An unordered list of **sequence-numbers**. See § 6.3.2.2.

8.5.2 *Delete-result*

Should the request succeed, the **delete-result** will be returned. There are no parameters.

```
DeleteResult ::= NULL
```

8.5.3 *Delete abstract-errors*

Should the request fail, one of the listed abstract-errors will be reported. The circumstances under which the particular abstract-errors will be reported are defined in § 9.

8.6 *Register-MS abstract-operation*

The Register-MS abstract-operation is used to register or deregister various information with the MS:

a) auto-actions;

b) default list of attribute-types;

c) new credentials;

d) new set of user-security labels.

```
Register-MS ::= ABSTRACT-OPERATION
    ARGUMENT Register-MSArgument
    RESULT     Register-MSResult
    ERRORS{
        AtrributeError,
        AutoActionRequestError,
        InvalidParametersError,
        SecurityError,
        ServiceError}
```

8.6.1 *Register-MS-argument*

```
Register-MS Arguments := SET{
        auto-action-registrations      [0] SET SIZE (1 .. ub-auto-registrations) OF AutoActionRegistration OPTIONAL,
        auto-action-deregistrations    [1] SET SIZE (1 .. ub-auto-registrations) OF AutoActionDeregistration OPTIONAL,
        list-attribute-defaults        [2] SET SIZE (1 .. ub-default-registrations) OF Attribute Type OPTIONAL,
        fetch-attribute-defaults       [3] SET SIZE (1 .. ub-default-registrations) OF Attribute Type OPTIONAL,
        change-credentials             [4] SEQUENCE{
            old-credentials                [0] IMPLICIT Credentials,
            new-credentials                [1] IMPLICIT Credentials} OPTIONAL
              -- same CHOICE as for old-credentials --,
        user-security-labels           [5] SET SIZE (1 .. ub-labels-and-redirections) OF SecurityLabel OPTIONAL}
```

The components of register-MS-argument have the following meanings:

1) **Auto-action-registrations** (O): This is a set of **auto-action-registration** (see § 6.5.1), one for each auto-action to be registered. The new **auto-action-registration-parameter** supersedes any previously registered auto-action (if any) with that **registration-identifier** and **auto-action-type**.

   In the absence of **auto-action-registrations**, the default is that no new auto-actions are registered.

2) **Auto-action-deregistrations** (O): This is a set of **auto-action-deregistration**, one for each auto-action to be deregistered. Any auto-action with **registration-identifier** and **auto-action-type** matching those in an **auto-action-deregistration** is deregistered.

   AutoActionDeregistration ::= AutoActionRegistration
   (WITH COMPONENTS {..., registration-parameter ABSENT})

   In the absence of **auto-action-deregistrations**, the default is that no registered auto-actions are deregistered.

3) **List-attribute-defaults** (O): This specifies a default set of attribute-types to indicate which attributes should be returned for any subsequent list or alert abstract-operation if the entry-information-selection argument is absent.

   In the absence of **list-attribute-defaults**, the default is that there is no change to the registered default (if any). The **list-attribute-defaults** are the empty set until explicitly changed by the MS-user via the register-MS abstract-operation.

4) **Fetch-attribute-defaults** (O): This specifies a default set of attribute-types to indicate which attributes should be returned for any subsequent fetch abstract-operation if the entry-information-selection argument is absent.

   In the absence of **fetch-attribute-defaults**, the default is that there is no change to the registered default (if any). The **fetch-attribute-defaults** are the empty set until explicitly changed by the MS-user via the register-MS abstract-operation.

5) **Change-credentials** (O): The old and new credentials if a **change-credentials** is requested.

   The **old-credentials** are the end user's current credentials, and the **new-credentials** are the credentials the end user would like to change to.

   In the absence of this argument, the default is that previously registered credentials remain unchanged.

   The credentials of the MS-user may differ from the **initiator-credentials** detailed in § 8.1.1.1.1.2 of Recommendation X.411.

6) **User-security-labels** (O): This contains the **security-label(s)** of the MS abstract-service-user, if they are to be changed. It may be generated by the MS abstract-service-user.

   In the absence of this argument, the **user-security-labels** remain unchanged.

   Note that some **security-policies** may only permit the **user-security-labels** to be changed in this way if a secure link is employed. Other local means of changing the **user-security-labels** in a secure manner may be provided. **User-security-labels** is defined in § 8.4.1.1.1.7 of Recommendation X.411.

   Security-label is defined in § 9 of Recommendation X.411.

8.6.2 *Register-MS-result*

Should the request succeed, the register-MS-result will be returned. There are no parameters.

Register-MSResult ::= NULL

8.6.3 *Register-MS abstract-errors*

Should the request fail, one of the listed abstract-errors will be reported. The circumstances under which the particular abstract-errors will be reported are defined in § 9.

8.7 *Alert abstract-operation*

The **Alert abstract-operation** enables the MS abstract-service-provider to immediately inform the MS abstract-service-user of a new entry having been entered into the MS, whose attributes match the selection criteria of one of the **auto-alert-registrations** (see § 12.2) previously supplied using a Register-MS abstract-operation (see § 8.6).

The **Alert abstract-operation** may be invoked during an existing abstract-association initiated by the UA, and only as a result of new entries created after the establishment of the abstract-association.

Entries matching the selection criteria which have been created between abstract-associations will be indicated in the result of the next abstract-bind-operation for the abstract-association. No **alert abstract-operation** will be invoked for these entries. See § 7.

The **alert abstract-operation** will only be successful when the information-base permits access according to the security-context and the enforced security-policy.

```
Alert ::= ABSTRACT-OPERATION
          ARGUMENT        AlertArgument
          RESULT          AlertResult
          ERRORS {
              SecurityError }
```

8.7.1 *Alert-argument*

```
AlertArgument ::= SET {
          alert-registration-identifier    [0] INTEGER (1 .. ub-auto-actions),
          new-entry                        [2] EntryInformation OPTIONAL }
```

The components of the **alert-argument** have the following meanings:

1) **Alert-registration-identifier** (M): Identifies which of the **auto-alert-registrations** resulted in **the** alert (see §§ 6.4 and 12.2).
2) **New-entry** (O): This conveys the information from the new entry which was requested in the **auto-alert-registration-parameter** (see § 12.2). It is absent when the MS abstract-service-user did not specify an **auto-alert-registration-parameter.**

8.7.2 *Alert-result*

Should the request succeed, the alert-result will be returned.

```
AlertResult ::= NULL
```

8.7.3 *Alert abstract-errors*

Should the request fail, one of the listed abstract-errors will be reported. The circumstances under which the particular abstract-errors will be reported are defined in § 9.

9 Abstract-errors

This paragraph defines the following abstract-errors associated with using the abstract-operations at the retrieval port:

a) AttributeError;
b) AutoActionRequestError;
c) DeleteError;
d) FetchRestrictionError;
e) InvalidParametersError;
f) RangeError;
g) SecurityError;
h) SequenceNumberError;
i) ServiceError;

9.1 *Error precedence*

The performer of an abstract-operation is not required to continue processing the message beyond the point at which an error has been detected. This allows an implementation to choose whether to continue the processing of errors.

Note – An implication of this rule is that the first error encountered may differ for repeated instances of the same abstract-operation, as there is not necessarily a specific logical order in which to process it.

9.2 *Attribute-error*

An **attribute-error** reports an attribute related problem.

```
AttributeError ::= ABSTRACT-ERROR
        PARAMETER SET{
            problems [0] SET SIZE (1 .. ub-per-entry) OF SET{
                problem     [0] AttributeProblem,
                type        [1] AttributeType,
                value       [2] ANY DEFINED BY type OPTIONAL}}

AttributeProblem ::= INTEGER{
        invalid-attribute-value          (0),
        unavailable-attribute-type       (1),
        inappropriate-matching           (2),
        attribute-type-not-subscribed    (3),
        inappropriate-for-operation      (4)} (0 .. ub-error-reasons)
```

The parameter has the following meaning:

1) **Problems** (M): The particular problems encountered. Any numbers of individual problems may be indicated, each problem being accompanied by an indication of the attribute-type, and, if necessary to avoid ambiguity, the value which caused the problem:

   a) **Invalid-attribute-value** (C): A purported attribute-value specified as an argument of the abstract-operation does not conform to the data-type defined for the attribute-type concerned.

   b) **Unavailable-attribute-type** (C): A purported attribute-type used as an argument of the abstract-operation is not one of those which is supported by the MS abstract-service-provider. If the MS abstract-service-provider is able to carry out the operation anyway, it is not prohibited from doing so.

   c) **Inappropriate-matching** (C): The filter contains a filter-item in which an attribute is matched using an operation (equality, ordering, or substrings) that is not defined for that attribute.

   d) **Attribute-type-not-subscribed** (C): An attribute-type used as an argument of the abstract-operation is not one of those to which the MS abstract-service-user has subscribed.

      Note – A change of the subscription is not necessarily reflected in the attributes present in an entry created before the change.

   e) **Inappropriate-for-operation** (C): An attribute-type used as an argument of the abstract-operation is unsuitable for its required use.

9.3 *Auto-action-request-error*

An **auto-action-request-error** reports a problem related to registration of an auto-action.

```
AutoActionRequestError ::= ABSTRACT-ERROR
        PARAMETER SET{
            problems [0]        SET SIZE (1 .. ub-auto-registrations) OF SET{
                problem         [0] Auto-ActionRequestProblem,
                type            [1] AutoActionType}}

AutoActionRequestProblem ::= INTEGER{
        unavailable-auto-action-type      (0),
        auto-action-type-not-subscribed   (1)} (0 .. ub-error-reasons)
```

The parameter has the following meaning:

1) **Problems** (M): The particular problems encountered. Any numbers of individual problems may be indicated, each problem being accompanied by an indication of the **auto-action-type** which caused the problem:

   a) **Unavailable-auto-action-type**: An auto-action-type used as an argument of the abstract-operation is not one of those which is supported by the MS abstract-service-provider.

b) **Action-type-not-subscribed**: An action-type used as an argument of the abstract-operation is not one of those to which the MS abstract-service-user has subscribed.

9.4 *Delete-error*

A **delete-error** reports a problem in an attempt to delete one or more entries from an information-base.

```
DeleteError ::= ABSTRACT-ERROR
        PARAMETER SET{
            problems [0] SET SIZE (1 .. ub-messages) OF SET{
                problem          [0] DeleteProblem,
                sequence-number  [1] SequenceNumber}}

DeleteProblem ::= INTEGER{
        child-entry-specified          (0),
        delete-restriction-problem     (1)} (0 .. ub-error-reasons)
```

The parameter has the following meaning:

1) **Problems** (M): The particular problems encountered. Any number of individual problems may be indicated, each problem being accompanied by an indication of the sequence-number of the entry which caused the problem:

   a) **Child-entry-specified**: An attempt has been made to delete a child-entry.

   b) **Delete-restriction-problem**: An attempt has been made to violate a restriction specified for the Delete abstract-operation (see § 8.5).

9.5 *Fetch-restriction-error*

A **fetch-restriction-error** reports an attempt to violate a restriction associated with the fetch abstract-operation.

```
FetchRestrictionError ::= ABSTRACT-ERROR
        PARAMETER SET{
            problems [0] SET SIZE (1 .. ub-default-registrations) OF SET{
                problem          [3] FetchRestrictionProblem,
                restriction      CHOICE{
                  content-type       [0] ContentType,
                  eit                [1] MS-EITs,
                  content-length     [2] ContentLength}}}

FetchRestrictionProblem ::= INTEGER{
        content-type-problem           (1),
        eit-problem                    (2),
        content-length-problem         (3)} (0 .. ub-error-reasons)
```

The parameter has the following meaning:

1) **Problems** (M): The particular problems encountered. Any number of individual problems may be indicated, each problem being accompanied by an indication of the offending content-type, encoded-information-type or content-length which caused the problem:

   a) **Content-type-problem** (C): The content-type of the message being fetched is disallowed by the fetch-restrictions currently in force.

   b) **Eit-problem** (C): The encoded-information-types requested in the Fetch abstract-operation are disallowed by the fetch-restrictions currently in force.

   c) **Content-length-problem** (C): The content-length of the message being fetched is longer than that allowed by the fetch-restrictions currently in force.

9.6 *Invalid-parameters-error*

An **invalid-parameters-error** reports a semantic problem in the set of parameters received. This error would be used, for example, to report that an optional parameter was present in the wrong context, or to report that a value for one of the parameters is inappropriate.

```
InvalidParametersError ::= ABSTRACT-ERROR
        PARAMETER NULL
```

This error has no parameters.

9.7 *Range-error*

A **range-error** reports a problem related to the limit specified in a selector as an argument to an abstract-operation.

```
RangeError ::= ABSTRACT-ERROR
        PARAMETER SET {
            problem          [0] RangeProblem }

RangeProblem ::= INTEGER {
        reversed             (0) } (0 .. ub-error-reasons)
```

The parameter has the following meaning:

1) **Problems** (M): The particular problems encountered:
   a) **Reversed** (C): The upper bound indicated a sequence-number or creation-time before that indicated by the lower bound.

9.8 *Security-error*

A **security-error** reports that the requested abstract-operation cannot be provided because it would violate the security-policy in force. This error is defined in Recommendation X.411.

9.9 *Sequence-number-error*

A **SequenceNumberError** reports a problem related to the sequence-number specified in an argument to an abstract-operation.

```
SequenceNumberError ::= ABSTRACT-ERROR
        PARAMETER SET {
            problems [1] SET SIZE (1 .. ub-messages) OF SET {
                problem          [0] SequenceNumberProblem,
                sequence-number  [1] SequenceNumber }}

SequenceNumberProblem ::= INTEGER {
        no-such-entry            (0) } (0 .. ub-error-reasons)
```

The parameter has the following meaning:

1) **Problems** (M) : The particular problems encountered. Any numbers of individual problems may be indicated, each problem being accompanied by an indication of the sequence-numbers which caused the problem:
   a) **No-such-entry** : The sequence-number supplied does not match that of any entry in the information-base.

9.10 *Service-error*

A **service-error** reports an error related to the provision of the service.

```
ServiceError ::= ABSTRACT-ERROR
        PARAMETER SET {
            problem              [0] ServiceProblem }

ServiceProblem ::= INTEGER {
        busy                     (0),
        unavailable              (1),
        unwilling-to-perform     (2) } (0 .. ub-error-reasons)
```

The parameter has the following meaning:

1) **Problem** (M): The particular problem encountered:
   a) **Busy** (C): The MS, or some part of it, is presently too busy to perform the requested abstract-operation, but may be able to do so after a short while.
   b) **Unavailable** (C): The MS, or some part of it, is presently unavailable.
   c) **Unwilling-to-perform** (C): The MS is not prepared to execute this request, because it would lead to excessive consumption of resources.

10 Overview

The MS information-model and the **attribute** and **auto-action** concepts were introduced in § 6.3.3 and § 6.5. Paragraph 11 defines the **general-attribute-types** which are specified for MS. Paragraph 12 defines the **general-auto-action-types** which are specified for MS.

11 General-attribute-types

The **general-attribute-types** are valid for all message content-types. Other attribute-types, which are content-specific, are defined in their respective Recommendations, e.g. the IPMS-specific attribute-types for MS are defined in Annex C of Recommendation X.420.

11.1 *General-attribute-types overview*

The **general-attributes** that may occur in a stored-messages information-base entry are listed in Table 1/X.413. They are constructed mainly from the parameter information from the MessageDelivery and ReportDelivery abstract-operations of the MTS abstract-service as defined in § 8 of Recommendation X.411, and such attributes are correspondingly named. Some **general-attributes** are generated, and some of these also maintained, by the MS.

Table 1/X.413 defines the various **general-attributes** and defines the following for each attribute-type:

- whether the attribute-type is single-valued or multi-valued;
- whether or not support by the MS and the accessing UA is mandatory or optional;
- whether the attribute-type is always present, conditionally present, or absent in a delivered-message entry, a delivered-report entry, or a returned-content entry respectively;
- whether or not the attribute-type can be returned in a list or an alert abstract-operation;
- whether or not the attribute-type may be used in a summarize abstract-operation.

  Note – Only for simple ASN.1 data-types.

For a more detailed description of the classification in Table 1/X.413 refer to the conventions in § 5.2.

An optional attribute-type is only suported by an MS if the support of that attribute-type has successfully been subscribed to (which implies that the MS and the accessing UA supports that attribute). Subscription to optional attribute-types can be per attribute-type per UA.

11.2 *Description of the general-attribute-types*

The following paragraphs contain a short description of each **general-attribute-type** together with its abstract-syntax using the ATTRIBUTE macro described in § 6.3.

It should be noted that some **general-attributes** are used primarily for filtering and listing purposes while others can contain more complex (further structured ASN.1 data-types) and potentially voluminous information. Only a few **general-attributes** are suitable for summaries.

11.2.1 *Child-sequence-numbers*

This general-attribute, which is multi-valued, contains one or more "pointers" to the next level of child-entries, if such exist. It is generated by the MS. It is present in a parent-entry that has one or more child-entries associated with it. It is absent in an entry without child-entries.

```
ms-child-sequence-numbers ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX-SequenceNumber
        MULTI VALUE
        ::= id-att-child-sequence-numbers
```

TABLE 1/X.413

General-attribute-types for the stored-messages

Attribute-type-name	Single/ multi valued	Support level by MS and access UA	Presence in delivered message entry	Presence in delivered report entry	Presence in returned-content entry	Available for list, alert	Available for summarize
Child-sequence-numbers	M	M	C	C	C	Y	N
Content	S	M	P	–	P	N	N
Content-confidentiality-algorithm-identifier	S	O	C	–	–	Y	N
Content-correlator	S	O	–	C	–	Y	N
Content-identifier	S	O	C	C	–	Y	N
Content-integrity-check	S	O	C	–	–	Y	N
Content-length	S	O	P	–	P	Y	N
Content-returned	S	O	–	P	–	Y	Y
Content-type	S	M	P	C	C	Y	Y
Conversion-with-loss-prohibited	S	O	C	–	–	Y	N
Converted-EITs	M	O	C	–	–	Y	N
Creation-time	S	M	P	P	P	Y	N
Delivered-EITs	M	O	P	–		Y	N
Delivery-flags	S	O	P	–	–	Y	N
DL-expansion-history	M	O	C	C	–	Y	N
Entry-status	S	M	P	P	P	Y	Y
Entry-type	S	M	P	P	P	Y	Y
Intended-recipient-name	S	O	C	–	–	Y	N
Message-delivery-envelope	S	M	P	–	–	N	N
Message-delivery-identifier	S	O	P	–	–	Y	N
Message-delivery-time	S	O	P	–	–	Y	N
Message-origin-authentification-check	S	O	C	–	–	Y	N
Message-security-label	S	O	C	C	–	Y	N
Message-submission-time	S	O	P	–	–	Y	N
Message-token	S	O	C	–	–	Y	N
Original-EITs	M	O	C	C	–	Y	N
Originator-certificate	S	O	C	–	–	Y	N
Originator-name	S	O	P	–	–	Y	N
Other-recipient-names	M	O	C	–	–	Y	N
Parent-sequence-number	S	M	C	–	P	Y	N
Per-recipient-report-delivery-fields	M	M	–	P	–	Y	N
Priority	S	O	P	–	–	Y	Y
Proof-of-delivery-request	S	O	C	–	–	Y	N
Redirection-history	M	O	C	–	–	Y	N
Report-delivery-envelope	S	M	–	P	–	N	N
Reporting-DL-name	S	O	–	C	–	Y	N
Reporting-MTA-certificate	S	O	–	C	–	Y	N
Report-origin-authentification-check	S	O	C	C	–	Y	Y
Security-classification	S	O	C	C	–	Y	Y
Sequence-number	S	M	P	P	P	Y	N
Subject-submission-identifier	S	M	–	P	–	Y	N
This-recipient-name	S	O	P	–	–	Y	N

11.2.2 *Content*

This general-attribute contains the complete content of a message as delivered by the MessageDelivery abstract-operation or as returned-content by the ReportDelivery abstract-operation. For more details see §§ 8.2.1.1.1.37 and 8.3.1.2.1.14 of Recommendation X.411.

ms-contentATTRIBUTE
WITH ATTRIBUTE-SYNTAX Content
SINGLE VALUE
::= id-att-content

11.2.3 *Content-confidentiality-algorithm-identifier*

This general attribute contains the **algorithm-identifier** used by the originator of the message to encrypt the message content. It may be generated by the originator of the message. For further details see § 8.5.10 of Recommendation X.411.

mt-content-confidentiality-algorithm-identifier ATTRIBUTE
WITH ATTRIBUTE-SYNTAX AlgorithmIdentifier
SINGLE VALUE
::= id-att-content-confidentiality-algorithm-identifier

11.2.4 *Content-correlator*

This general-attribute contains information to enable correlation of the content of the message. It may be generated by the originating UA. For more details see § 8.2.1.1.1.36 of Recommendation X.411.

mt-content-correlator ATTRIBUTE
WITH ATTRIBUTE-SYNTAX ContentCorrelator
MATCHES FOR EQUALITY
SINGLE VALUE
::= id-att-content-correlator

11.2.5 *Content-identifier*

This general-attribute contains an identifier for the content of the message. It may be generated by the originating UA. For more details see § 8.2.1.1.1.35 of Recommendation X.411.

mt-content-identifier ATTRIBUTE
WITH ATTRIBUTE-SYNTAX ContentIdentifier
MATCHES FOR EQUALITY
SINGLE VALUE
::= id-att-content-identifier

11.2.6 *Content-integrity-check*

This general attribute provides the recipient(s) of the message with a means of validating that the message content has not been modified. It may be generated by the originator of the message and may specify a different value for each recipient of the message. For further details see § 8.2.1.1.28 of Recommendation X.411.

mt-content-integrity-check ATTRIBUTE
WITH ATTRIBUTE-SYNTAX ContentIntegrityCheck
SINGLE VALUE
::= id-att-content-integrity-check

11.2.7 *Content-length*

This general-attribute gives the length of the content in octets of a message as delivered by the MessageDelivery abstract-operation or of a returned-content (if any) notified by the ReportDelivery abstract-operation. Where there is no such returned-content, this attribute is absent. It is generated by the MS.

ms-content ATTRIBUTE
WITH ATTRIBUTE-SYNTAX ContentLength
MATCHES FOR ORDERING
SINGLE VALUE
::= id-att-content-length

11.2.8 *Content-returned*

This general-attribute indicates whether a content has been returned in the ReportDelivery abstract-operation. It is generated by the MS.

```
ms-content-returned ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX BOOLEAN
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-content-returned
```

11.2.9 *Content-type*

This general-attribute is generated from the content-type in the MessageDelivery or ReportDelivery abstract-operation. See also § 8.2.1.1.1.34 of Recommendation X.411.

```
mt-content-type ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX OBJECT IDENTIFIER
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-content-type
```

11.2.10 *Conversion-with-loss-prohibited*

This general-attribute contains information about whether conversion with loss of information was allowed or prohibited. For further details see § 8.2.1.1.1.10 of Recommendation X.411.

```
mt-conversion-with-loss-prohibited ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ConversionWithLossProhibited
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-conversion-with-loss-prohibited
```

11.2.11 *Converted-EITs*

This general-attribute, which is multi-valued, identifies the encoded-information-types of the content after conversion, as indicated by MessageDelivery or ReportDelivery abstract-operation. It is generated by the MS. It is absent if no conversion took place. For more details see § 8.3.1.1.1.8 and 8.3.1.2.1.5 of Recommendation X.411.

```
ms-converted-EITs ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MS-EIT
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-att-converted-EITs
```

11.2.12 *Creation-time*

This general-attribute gives the time when the entry was created in the MS. It is generated by the MS. For more details see § 6.3.2.

Note – Two or more consecutive entries may have the same creation-time.

```
ms-creation-time ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX CreationTime
        MATCHES FOR EQUALITY ORDERING
        SINGLE VALUE
        ::= id-att-creation-time
```

11.2.13 *Delivered-EITs*

This general-attribute, which is multi-valued, identifies the encoded-information-types in the content of the message as delivered. It is generated by the MS based on information about the original-EITSs and the converted-EITs in the MessageDelivery abstract-operation.

```
ms-delivered-EITs ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MS-EIT
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-att-delivered-EITs
```

11.2.14 *Delivery-flags*

This general-attribute contains information of the delivery. Presently, it is only used for indicating implicit-conversion of the content. For more details see § 8.2.1.1.1.9 of Recommendation X.411.

```
mt-delivery-flags ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX DeliveryFlags
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-delivery-flags
```

11.2.15 *DL-expansion-history*

This general-attribute, which is multi-valued, is used to show the history of distribution-list expansion. It contains one or more distribution-list names used during the expansion process. It is absent if the delivery to this recipient did not involve any expansion of a distribution-list. For more details see § 8.3.1.1.1.7 of Recommendation X.411.

```
mt-dl-expansion-history ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX DLExpansionHistory
        MULTI VALUE
        ::= id-att-dl-expansion-history
```

11.2.16 *Entry-status*

This general-attribute contains the current status of an entry in the stored-messages information-base. It is created and maintained by he MS. For more details see § 6.4.

```
ms-entry-status ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX EntryStatus
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-entry-status
```

11.2.17 *Entry-type*

This general-attribute contains information about whether an entry concerns a delivered message or a delivered report. It is generated by the MS.

```
ms-entry-type ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX EntryType
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-entry-type

EntryType ::= INTEGER {
        delivered-message       (0),
        delivered-report        (1),
        returned-content        (2) (0 .. ub-entry-types) }
```

11.2.18 *Intended-recipient-name*

This general-attribute contains the O/R-name of the originally intended recipient if the message has been redirected, with each value representing one redirection. For more details see § 8.3.1.1.1.4 of Recommendation X.411.

```
mt-intended-recipient-name ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ORName
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-intended-recipient-name
```

11.2.19 *Message-delivery-envelope*

This general-attribute contains the complete **message-delivery-envelope** of a message as delivered by the MessageDelivery abstract-operation. For more details see § 9 of Recommendation X.411.

```
mt-message-delivery-envelope ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MessageDeliveryEnvelope
        SINGLE VALUE
        ::= id-att-message-delivery-envelope
```

11.2.20 *Message-delivery-identifier*

This general-attribute contains the **message-delivery-identifier** from the MessageDelivery abstract-operation. For more details see § 8.3.1.1.1.1 of Recommendation X.411.

```
mt-message-delivery-identifier ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MessageDeliveryIdentifier
        SINGLE VALUE
        ::= id-att-message-delivery-identifier
```

11.2.21 *Message-delivery-time*

This general-attribute contains the **message-delivery-time** from the MessageDelivery abstract-operation. For more details see § 8.3.1.1.1.2 of Recommendation X.411.

Note – There is no general-attribute corresponding to the delivery-time parameter of the ReportDelivery abstract-operation, because in order to be useful, this delivery-time must be correlated with the name of the recipient the message was delivered to. This information is included in the report-information general-attribute.

```
mt-message-delivery-time ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MessageDeliveryTime
        MATCHES FOR EQUALITY ORDERING
        SINGLE VALUE
        ::= id-att-message-delivery-time
```

11.2.22 *Message-origin-authentication-check*

This general attribute is computed using the algorithm identified by the message-origin-authentication-identifier. It provides the recipient(s) of the message with a means of authenticating the origin of the message and may be generated by the originator of the message. For further details see § 8.2.1.1.1.29 of Recommendation X.411.

```
mt-message-origin-authentication-check ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MessageOriginAuthenticationCheck
        SINGLE VALUE
        ::= id-att-message-origin-authentication-check
```

11.2.23 *Message-security-label*

This general attribute comprises a set of security attributes which may include a security-policy-identifier, a security-classification, a privacy-mark, and a set of security-categories. For further details see § 8.2.1.1.1.30 of Recommendation X.411.

```
mt-message-security-label ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MessageSecurityLabel
        SINGLE VALUE
        ::= id-att-message-security-label
```

11.2.24 *Message-submission-time*

This general-attribute contains the **message-submission-time** from a MessageDelivery abstract-operation. For more details see § 8.2.1.1.2.2 of Recommendation X.411.

```
mt-message-submission-time ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MessageSubmissionTime
        MATCHES FOR EQUALITY ORDERING
        SINGLE VALUE
        ::= id-att-message-submission-time
```

11.2.25 *Message-token*

This general attribute contains the token associated with the message. It is generated by the originator of the message and may contain a different value for each recipient of the message. For further details see § 8.2.1.1.1.26 of Recommendation X.411.

```
mt-message-token ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MessageToken
        SINGLE VALUE
        ::= id-att-message-token
```

11.2.26 *Original-EITs*

This general-attribute, which is multi-valued, contains the **orginal encoded-information-types** from the MessageDelivery abstract-operation. It is generated by the MS. For more details see § 8.2.1.1.1.33 of Recommendation X.411.

```
ms-original-EITs ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MS-EIT
        MATCHES FOR EQUALITY
        MULTI VALUE
    ::= id-att-original-EITs
```

11.2.27 *Originator-certificate*

This general attribute, contains the certificate of the originator of the message. It is generated by a trusted source (e.g. a certification-authority), and may be supplied by the originator of the message. For further details see § 8.2.1.1.1.25 of Recommendation X.411.

```
mt-originator-certificate ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX OriginatorCertificate
        SINGLE VALUE
        ::= id-att-originator-certificate
```

11.2.28 *Originator-name*

This general-attribute contains the O/R-name of the originator from the MessageDelivery abstract-operation. For more details see § 8.2.1.1.1.1 of Recommendation X.411.

```
mt-originator-name ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ORName
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-originator-name
```

11.2.29 *Other-recipient-names*

This general-attribute, which is multi-valued, contains the O/R-names of all other specified recipients, if any, of the message from the MessageDelivery abstract-operation. For more details see § 8.3.1.1.1.6 of Recommendation X.411.

```
mt-other-recipient-names ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ORName
        MATCHES FOR EQUALITY
        MUTLI VALUE
        ::= id-att-other-recipient-names
```

11.2.30 *Parent-sequence-number*

This general-attribute, points to a parent-entry. It is generated by the MS. It is always present in a child-entry and is absent in a main-entry.

```
ms-parent-sequence-number ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX SequenceNumber
        MATCHES FOR EQUALITY ORDERING
        SINGLE VALUE
        ::= id-att-parent-sequence-number
```

11.2.31 *Per-recipient-report-delivery-fields*

This general-attribute, which is multi-valued, contains report information on a per-recipient basis from the ReportDelivery abstract-operation. For more details see § 8.3.1.2 of Recommendation X.411.

```
mt-per-recipient-report-delivery-fields ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX PerRecipientReportDeliveryFields
        MUTLI VALUE
        ::= id-att-per-recipient-report-delivery-fields
```

11.2.32 *Priority*

This general-attribute contains the relative **priority** of the message from the MessageDelivery abstract-operation. If no value is supplied in the MessageDelivery abstract-operation parameter, the MS uses its default value when generating this attribute. For more details see § 8.2.1.1.1.8 of Recommendation X.411.

```
mt-priority ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX Priority
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-priority
```

11.2.33 *Proof-of-delivery-request*

This general attribute indicates whether or not the originator of the message requires **proof-of-delivery** of the message to the recipient. It may be generated by the originator of the message and may specify a different value for each recipient of the message. For more details see § 8.2.1.1.1.32 of Recommendation X.411.

```
mt-proof-of-delivery-request ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ProofOfDeliveryRequest
        SINGLE VALUE
        ::= id-att-proof-of-delivery-request
```

11.2.34 *Redirection-history*

The general-attribute, which is multi-valued, contains the history of recipient redirection(s) with reasons(s) from the MessageDelivery or ReportDelivery abstract-operation. For more details see § 8.3.1.1.1.5 of Recommendation X.411.

```
mt-redirection-history ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX RedirectionHistory
        MULTI VALUE
        ::= id-att-redirection-history
```

11.2.35 *Report-delivery-envelope*

This general-attribute contains all the parameters from the ReportDelivery abstract-operation, except for the returned-content (if present). For more details see § 8.3.1.2 of Recommendation X.411.

```
mt-report-delivery-envelope ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ReportDeliveryEnvelope
        SINGLE VALUE
        ::= id-att-report-delivery-envelope
```

11.2.36 *Reporting-DL-name*

This general-attribute contains the O/R-name of the distribution-list that forwarded the report to the owner of this distribution-list. For more details see § 8.3.1.2.1.4 of Recommendation X.411.

```
mt-reporting-DL-name ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ReportingDLName
        SINGLE VALUE
        ::= id-att-reporting-DL-name
```

11.2.37 *Reporting-MTA-certificate*

This general-attribute contains the certificate of the MTA that generated the report. For more details see § 8.3.1.2.1.12 of Recommendation X.411.

```
mt-reporting-MTA-certificate-ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ReportingMTACertificate
        SINGLE VALUE
        ::= id-att-reporting-MTA-certificate
```

11.2.38 *Report-origin-authentication-check*

The general-attribute provides a means of authenticating the origin of the report. For more details see § 8.3.1.2.1.13 of Recommendation X.411.

```
mt-report-origin-authentication-check ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ReportOriginAuthenticationCheck
        SINGLE VALUE
        ::= id-att-report-origin-authentication-check
```

11.2.39 *Security-classification*

This general-attribute comprises the security-classification parameter from the message-security-label. It is defined as a separate attribute to allow its use in the Summarize abstract-operation. For more details see § 8.5.9 of Recommendation X.411.

```
mt-security-classification ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX SecurityClassification
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-security-classification
```

11.2.40 *Sequence-number*

This general-attribute is used to identify the entry itself. It is allocated by the MS when the entry is created. For more details see § 6.3.2.

```
ms-sequence-number ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX SequenceNumber
        MATCHES FOR EQUALITY ORDERING
        SINGLE VALUE
        ::= id-att-sequence-number
```

11.2.41 *Subject-submission-identifier*

This general-attribute contains the message-submission-identifier or the probe-submission-identifier of the subject of the report. For more details see § 8.3.1.2.1.1 of Recommendation X.411.

```
mt-subject-submission-identifier ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX SubjectSubmissionIdentifier
        SINGLE VALUE
        ::= id-att-subject-submission-identifier
```

11.2.42 *This-recipient-name*

This general-attribute contains the O/R-name of this (MS) recipient from the MessageDelivery abstract-operation. For more details see § 8.3.1.1.1.3 of Recommendation X.411.

```
mt-this-recipient-name ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ORName
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-this-recipient-name
```

11.3 *Generation of the general-attributes*

This section describes how the general-attributes are generated. The information is presented in Table 2/X.413. For a description of the classification used, see § 5.3.

TABLE 2/X.413

Generation of the general-attribute-types

Attribute-type-name	Single/ multi valued	Source parameter	Source generated by	Generation rules
Child-sequence-numbers	M	–	MS	A value is generated for each corresponding child-entry that a parent-entry has
Content	S	**content** **returned-content**	MD RD	The value of the parameter is the attribute-value
Content-confidentiality-algorithm-identifier	S	**content-confidentiality-algorithm-identifier**	MD	The value of the parameter is the attribute-value
Content-correlator	S	**content-correlator**	RD	The value of the parameter is the attribute-value
Content-identifier	S	**content-identifier**	MD RD	The value of the parameter is the attribute-value
Content-integrity-check	S	**content-integrity-check**	MD	The value of the parameter is the attribute-value
Content-length	S	–	MS	The (approximate) size of the stored content in octets based on the delivered or returned content
Content-returned	S	–	MS	The value is set to **true** if returned-content is present in a ReportDelivery and to **false** if not present
Content-type	S	**content-type**	MD RD	If represented by OBJECT IDENTIFIER, the value of the parameter. If represented by INTEGER, converted to the corresponding OBJECT IDENTIFIER
Conversion-with-loss-prohibited	S	**conversion-with-loss-prohibited**	MD	The value of the parameter is the attribute-value
Converted-EITs	M	**converted-encoded-information-types**	MD	A corresponding value is generated from each bit that is set to **1** in the **built-in-encoded-information-types** parameter and from each **ExternalEncoded InformationType** present in the **external-encoded-information-type** parameter
Creation-time	S	–	MS	The time of creation of the entry
Delivered-EITs	M	**converted-EITs and original-EITs**	MS	A union of the other two general-attribute-types

TABLE 2/X.413 *(cont.)*

Attribute-type-name	Single/ multi valued	Source parameter	Source generated by	Generation rules
Delivery-flags	S	**delivery-flags**	MD	The value of the parameter is the attribute-value. If there are no **delivery-flags** in the MD, generate a default value with no flags set
DL-expansion-history	M	**DL-expansion-history**	MD RD	A corresponding value is generated from each component of the SEQUENCE
Entry-status	S	–	MS	Generated when the entry is created with the value "new"
Entry-type	S	**MessageDelivery ARGUMENT** **ReportDelivery ARGUMENT**	MS MS	The value is set to "delivered-message" The value is set to "delivered-report". If a returned-content is present, a child-entry, which contains the returned-content is created. For the child-entry, the value of this attribute is set to "returned-content"
Intended-recipient-name	S	**intended-recipient-name**	MD	The value of the parameter is the attribute-value
Message-delivery-envelope	S	**envelope**	MD	The value of the parameter is the attribute-value
Message-delivery-time	S	**message-delivery-time**	MD	The value of the parameter is the attribute-value
Message-delivery-identifier	S	**message-delivery-identifier**	MD	The value of the parameter is the attribute-value
Message-origin-authentication-check	S	**message-origin-authentication-check**	MD	The value of the parameter is the attribute-value
Message-security-label	S	**message-security-label**	MD RD	The value of the parameter is the attribute-value
Message-submission-time	S	**message-submission-time**	MD	The value of the parameter is the attribute-value
Message-token	S	**message-token**	MD	The value of the parameter is the attribute-value
Original-EITs	M	**original-encoded-information-types**	MD RD	A corresponding value is generated from each bit that is set to **1** in the **built-in-encoded-information-types** parameter and from each **ExternalEncoded InformationType** present in the **external-encoded-information-type** parameter
Originator-certificate	S	**originator-certificate**	MD	The value of the parameter is the attribute-value

TABLE 2/X.413 *(end)*

Attribute-type-name	Single/ multi valued	Source parameter	Source generated by	Generation rules
Originator-name	S	**originator-name**	MD	The value of the parameter is the attribute-value
Other-recipient-names	M	**other-recipient-names**	MD	A corresponding value is generated from each component of the SEQUENCE
Parent-sequence-number	S	–	MS	When creating a child-entry, this attribute is generated with the corresponding parent-entry's sequence-number as value
Per-recipient-report-delivery-fields	M	**per-recipient-fields**	RD	A corresponding value is generated from each component of the SEQUENCE
Priority	S	**priority**	MD	The value of the parameter is the attribute-value
Proof-of-delivery-request	S	**proof-of-delivery-request**	MD	The value of the parameter is the attribute-value
Redirection-history	M	**redirection-history**	MD	A corresponding value is generated from each component of the SEQUENCE
Report-delivery-envelope	S	**envelope**	RD	The value of the parameter is the attribute-value
Reporting-DL-name	S	**reporting-DL-name**	RD	The value of the parameter is the attribute-value
Reporting-MTA-certificate	S	**reporting-MTA-certificate**	RD	The value of the parameter is the attribute-value
Report-origin-authentication-check	S	**report-origin-authentication-check**	RD	The value of the parameter is the attribute-value
Security-classification	S	**security-classification**	MD RD	The value of the parameter is the attribute-value
Sequence-number	S	–	MS	When creating an entry, the MS assigns a unique value for this attribute in ascending order
Subject-submission-identifier	S	**subject-submission-identifier**	RD	The value of the parameter is the attribute-value
This-recipient-name	S	**this-recipient-name**	MD	The value of the parameter is the attribute-value

Note – When a message-delivery entry is created, there are no separate general-attributes generated for physical delivery and delivery method arguments, because the information in these arguments are not relevant to the MS. However, the UA can retrieve all the information contained in these arguments by retrieving the message-delivery-envelope general-attribute.

11.4 *Attribute-types subscription*

Attribute-type subscription is a local matter. If the attribute-type subscription is changed, then the UA may receive all of the attributes in the original subscription for messages present in the MS at the time the subscription was changed. The handling of these unsubscribed attributes is a local matter. Similarly, when a new attribute is subscribed to, the UA may not receive this attribute for messages in the MS when the subscription occurred.

12 General-auto-action-types

The **general-auto-action-types** are valid for all message content-types. However, their detailed effect may be content-specific, and so the procedure descriptions given in this Recommendation may need to be supplemented in their respective Recommendations, e.g. the IPMS-specific procedure for the auto-forward **general-auto-action-type** is described in § 19.4 of Recommendation X.420. Other **auto-action-types**, which are content-specific, may be defined in their respective Recommendations.

Auto-actions are introduced in § 6.5 and are registered and deregistered using the Register-MS abstract-operation described in § 8.6.

The following **general-auto-action-types** are defined:

a) Auto-forward;

b) Auto-alert.

The operation of **auto-actions** may be affected by the implementation of a security-policy.

The following subclauses contain a short description of each **general-auto-action-type** together with its abstract-syntax using the **AUTO-ACTION** macro defined in § 6.5.

12.1 *Auto-forward*

The **auto-forward auto-action** enables the MS abstract-service-provider to automatically **forward** any message that has been delivered into the stored-messages information base. The exact definition of forwarding is content-specific, but it always involves the submission of a new message incorporating the delivered content to the MTS abstract-service.

The **auto-forward auto-action-type** allows one or more sets of **auto-forward** parameters to be registered with the MS, each identified by its **auto-forward-registration-identifier**. Each **auto-forward-registration-parameter** specifies criteria to determine whether it applies to a particular delivered message, and if so a copy of the message is **auto-forwarded** using the Message-submission abstract-operation. That is to say, if a message matches more than one set of criteria, the message is **auto-forwarded** that many times.

The **auto-forward-registration-parameter** specifies whether the main-entry (and any associated child-entries) corresponding to the message is to be deleted after **auto-forwarding**. If any of the parameters acted upon indicates no-deletion (or if any of the submissions fail), then the entry is not deleted.

```
auto-forward AUTO-ACTION
        REGISTRATION PARAMETER IS AutoForwardRegistrationParameter
        ::= id-act-auto-forward

AutoForwardRegistrationParameter ::= SET{
        filter                      [0] Filter OPTIONAL,
        auto-forward-arguments      [1] AutoForwardArguments,
        delete-after-auto-forwarding [2] BOOLEAN DEFAULT FALSE,
        other-parameters            [3] OCTET STRING OPTIONAL}

AutoForwardArguments ::= SET{
        COMPONENTS OF PerMessageAutoForwardFields,
        per-recipient-fields        [1] IMPLICIT SEQUENCE (1..ub-recipients) OF
                                            PerRecipientAutoForwardFields}
```

```
PerMessageAutoForwardFields ::= SET{
        originator-name             OriginatorName,
        content-identifier          ContentIdentifier OPTIONAL,
        priority                    Priority DEFAULT normal,
        per-message-indicators      PerMessageIndicators DEFAULT {},
        deferred-delivery-time      [0] IMPLICIT DeferredDeliveryTime OPTIONAL,
        extensions                  [2] IMPLICIT PerMessageSubmissionExtensions DEFAULT {}}

PerRecipientAutoForwardFields ::= SET{
        recipient-name              RecipientName,
        originator-report-request   [0] IMPLICIT OriginatorReportRequest,
        explicit-conversion         [1] IMPLICIT ExplicitConversion OPTIONAL,
        extentions                  [2] IMPLICIT PerRecipientMessageSubmissionExtensions
                                        DEFAULT {}}
```

The parameters of the **auto-forward-registration-parameter** have the following meanings:

1) **Filter** (O): This is a set of criteria which a new entry representing a delivered message must satisfy for the MS abstract-service-provider to **auto-forward** it using this set of parameters.

   The absence of this parameter indicates that all new entries are **auto-forwarded.**

2) **Auto-forward-arguments** (M): This is a set of arguments registered to be used for each Message-submission abstract operation (see § 8.2.1.1.1 of Recommendation X.411). Any argument which is not registered, not mandatory, and not specifically mentioned below, will be absent from each Message-submission.

   If the following arguments are either not registered, or registered with their default values, the values used for each Message-submission abstract-operation are those of the corresponding message-delivery arguments: **priority, implicit-conversion-prohibited**, and **conversion-with-loss prohibited.**

   If the following arguments are either not registered, or registered with their default values, their presence as Message-submission arguments depends upon the presence of the corresponding Message-delivery arguments, their values being transformed where appropriate: **message-token, content-confidentiality-algorithm-identifier, content-integrity-check, message-origin-authentication-check**, and **message-security-label.**

   Certain Message-submission arguments may be registered. These are: **proof-of-submission-request, original-encoded-information-types, content-type**, and **content.**

3) **Delete-after-auto-forwarding** (O): This indicates whether an entry should be deleted or not, once the submission has succeeded.

   The absence of this parameter indicates that the message should not be deleted.

4) **Other-parameters** (O): This content-specific parameter need not be present. When it is present, the information it contains will be used during the **auto-forwarding** procedure.

   Note – Thus, for example, with Interpersonal Messaging, this parameter may contain the **auto-forward-comment** that is returned in the non-receipt notification, a user specified prefix and a cover-note accompanying the IP-message being auto-forwarded. For a description of **auto-forward-comment** usage, see § 19.4 of Recommendation X.420.

12.2 *Auto-alert*

The **auto-alert auto-action** enables the MS abstract-service-provider to automatically *alert* the user behind the MS abstract-service-user of the delivery of any message that has been delivered into the stored-messages information-base. **Auto-alert** will only be performed for delivered-message entries.

The **auto-alert auto-action-type** allows one or more sets of **auto-alert** parameters to be registered with the MS, each identified by its **auto-alert-registration-identifier.** Each **auto-alert-registration-parameter** specifies criteria to determine whether it applies to a particular delivered message. If a message matched the filter of more than one auto-alert-registration, the matching registration with the lowest auto-alert-registration-identifier is processed, and if at least one address (or the UA) has been alerted successfully, no other registrations are processed. If none of these addresses can be successfully alerted, the auto-alert registration with the next higher identifier is processed. This continues until either at least one or more addresses of a registration has been successfully alerted or the list of registrations is exhausted.

The **alert abstract-operation** will only be invoked if the alert-addresses in the auto-alert-registration is considered to have the UA as a member [see step 2) below]. If this alert-abstract-operation succeeds, any other address contained in the auto-alert registration will not be alerted.

```
Auto-alert AUTO-ACTION
        REGISTRATION PARAMETER IS AutoAlertRegistrationParameter
        ::= id-act-auto-alert

AutoAlertRegistrationParameter ::= SET{
        filter                  [0] Filter OPTIONAL,
        alert-addresses         [1] SEQUENCE SIZE (1..ub-alert-addresses) OF AlertAddress
                                    OPTIONAL,
        requested-attributes    [2] EntryInformationSelection OPTIONAL}
```

The parameters of the **auto-alert-registration-parameter** have the following meanings:

1) **Filter** (O): This is a set of criteria which a new entry representing a delivered-message must satisfy for the MS abstract-service-provider to **auto-alert** it using this set of parameters.

   The absence of this parameter indicates that **auto-alert** will be performed for all new delivered-message entries.

2) **Alert-addresses** (O): This argument identifies the types of **alert** service to be invoked, together with any information required to access the particular instances of those **alert** services, and any further information that needs to be conveyed during those **alerts**.

   Absence of this argument will default the alert abstract-operation to informing the MS abstract-service-user of the existence of an alert-condition either by using the alert abstract-operation (see § 8.7), (which is only possible if an abstract-association already exists between the MS abstract-service-user and the MS abstract-service-provider) or by flagging in the abstract-bind-operation next time the MS abstract-service-user establishes an abstract-association (see § 7). If the parameter requested-attributes is present, the MS abstract-service-user (UA) will be considered as being among the addresses to be alerted.

   Some types of **alert** will be internationally standardized. Others will be defined by national administrative authorities and private organizations. This implies that a number of separate authorities will be responsible for assigning types in a way that ensures that each is distinct from all other assigned types. This is accomplished by identifying each type with an object-identifier when the type is defined, and defining the ASN.1 data-type of the auxiliary addressing information.

   The **alert-qualifier** contains any further information that needs to be conveyed during the **auto-alert**. Absence of this parameter means that no additional information will be conveyed to the MS abstract-service-user.

```
AlertAddress ::= SEQUENCE{
    address             EXTERNAL,
    alert-qualifier     OCTET STRING OPTIONAL}
```

3) **Requested-attributes** (O): This indicates what information from the selected entries is to be included with the auto-alert. See § 8.1.4.

   The absence of this parameter implies that only the **alert-registration-identifier** will be present in the **alert-argument**.

13 Overview

This paragraph contains the procedures for the MS and the port realization. It contains a description of the consumption of the MTS abstract-service in § 14. The provision of the MS abstract-service is described in § 15. The port realization in the form of service elements is described in § 16.

The performance of the abstract-operations described in §§ 14 and 15 shall be subject to the requirements of the security-policy (if one is in force), which applies to the MTS abstract-services and to the MS abstract-services.

14 Consumption of the message transfer abstract-service

This paragraph specifies how an MS shall consume the MTS abstract-service which is defined in § 8 of Recommendation X.411. Covered are its consumption of the MTS delivery, submission, and administration ports.

14.1 *Consumption of the delivery port abstract-services*

This paragraph covers the performance of the MessageDelivery and ReportDelivery abstract-operations, and the invocation of the DeliveryControl abstract-operation. The MS consumption of the DeliveryPort abstract-services assumes that an abstract-association exists between the DeliveryPort supplier (the MTA) and the DeliveryPort consumer (the MS). The performance of the abstract-operations is in sequential order; no parallel processing takes place. Error cases are not described.

14.1.1 *Performance of the MessageDelivery abstract-operation*

When the MS receives a MessageDelivery abstract-operation from the MTA, it performs the following steps:

1) Returns a MessageDelivery result to the MTA to indicate that the delivery was successful. The MessageDelivery result shall contain proof-of-delivery information if the delivered-message contains a proof-of-delivery-request argument. This proof-of-delivery may be computed using the subject-MS-secret key; for more details see § 8.5.7 and § 8.3.1.1.2.2 of Recommendation X.411.

2) The next step is to examine if any auto-actions are activated. The auto-actions are partly content-specific and are therefore also described in the content-specific Recommendations. The content-specific description must contain rules about the order in which the auto-actions are to be performed. The performance of auto-actions may result in alerts, submissions, new entries being created and in the possible deletion of the delivered-message or other messages from the MS. See § 12.1.

   a) If auto-forwarding criteria are registered by the Register-MS abstract-operation, the new entry is matched against the criteria specified. The matching is made sequentially for each specified set of selection criteria. For every "hit" a new message is generated and submitted from the MS to the MTA using the MessageSubmission abstract-operation. See § 15.2.1.

      The rules for how to construct the new forwarded message are again content-specific and hence described in the respective content-specific Recommendations. Other content-specific events may also be performed at this stage (e.g. suppression of looping of auto-forwarded messages and the issuing of a non-receipt-notification as described for IPMS in § 19.4 of Recommendation X.420). Depending on the argument-values of the Register-MS abstract-operation for auto-forwarding, a copy of the delivered-message may be retained in the MS. If the auto-forwarding attempt is unsuccessful, a copy is always retained, to prevent messages from getting lost.

      Note – The handling of a result or error from such a submission is a local matter.

   b) If auto-alert-registrations have been made via the register-MS abstract-operation, the new entry is matched against the filter of each registration specified. The matching is made sequentially for each registration. If a "hit" is found, an attempt is made to invoke an alert abstract-operation from the MS to the UA. This can only be done if there is an existing abstract-association between the MS and the UA. If no abstract-association exists, the MS may have other local or non-standardized means to invoke an alert. When attempts have been made to alert all of the

addresses registered for the first matching registration parameter, and at least one of the alerts succeeded, the alert auto-action has successfully completed, and no further alert registrations are processed. If there was no path found to give the alert, the MS sets the alert-flag, which is reported to the UA when an abstract-association is next time initiated by the UA to the MS.

Note – If the delivered-message was deleted as a result of an auto-forwarding in a), the auto-alert is obviously not performed.

3) Only after the above steps have been performed is the new entry made visible outside the MS over the retrieval port. If the delivered-message was deleted as a result of an auto-action, any sequence-number which was allocated in step 2) is not re-used (in order not to conflict with ISO logging extensions). The entry-status of the entry is set to new.

14.1.1.1 *Generation rules for general-attributes*

Optional attributes are only generated if implemented by the MS and subscribed to by the user. The generated attributes form a new entry (in some cases a parent-entry and child-entries, see § 6) in the MS.

Refer to Table 1/X.413 and § 11.3 for the rules on how the general-attributes are generated. Note that for general-attributes which are absent in the corresponding deliver-envelope, an attribute with the default value is generated in the entry.

14.1.2 *Performance of the ReportDelivery abstract-operation*

When the MS receives a ReportDelivery abstract-operation from the MTA, it performs the following steps:

1) Returns a ReportDelivery result to the MTA to indicate that the delivery was successful. The ReportDelivery result has no parameters. For details, see § 8.3.1.2.2 of Recommendation X.411.
2) Next, if any auto-actions or other internal procedures are activated, they are performed. These are content-specific and described in the respective content-specific Recommendations.

14.1.2.1 *Generation rules for general-attributes*

Attributes may be generated either when a message is received or when an abstract-operation is performed in the MS, triggered by an invocation from the UA.

All mandatory attributes (see Table 1/X.413) are generated. Optional attributes are only generated if implemented by the MS and subscribed to by the user. The generated attributes form a new entry (in some cases a parent-entry and child-entries, see § 6) in the MS. The following kinds of general-attributes may be produced as part of the process:

a) general-attributes generated by the MS itself (e.g., sequence-number);
b) general-attributes generated from the report-delivery-envelope components. For components which are not present, but for which default values are defined, a general-attribute containing the default value is generated.

The generation rules for a) and b) are described in § 14.1.1.1. The generation rules for content-specific attributes are described in the respective content-specific Recommendations, e.g. the IPMS-specific attributes are described in Annex C of Recommendation X.420.

Refer to Table 1/X.413 and § 11.3 for the rules on how the general-attributes are generated. Note that for general-attributes which are absent in the corresponding report-envelope, an attribute with the default value is generated in the entry.

14.1.3 *Invocation of the DeliveryControl abstract-operation*

If the MS wants to temporarily stop the MTA from passing messages and reports, or to alter the maximum-content-length or lowest-priority of messages and reports from the MTA, it performs the following steps:

1) It invokes a DeliveryControl abstract-operation, containing the parameters to be changed. For details, see § 8.3.1.3 of Recommendation X.411.
2) It gets a result back when the MTS abstract-service has accepted the changes. The result contains information about whether messages and/or reports are waiting in the MTA, due to the current restrictions. For details, see § 8.3.1.3.2 of Recommendation X.411.

3) When the MS is able to accept any waiting messages and/or reports again, it should invoke a new DeliveryControl abstract-operation to relax the restrictions. The effects of a DeliveryControl abstract-operation are cancelled when either a new DeliveryControl abstract-operation alters the restrictions or when the abstract-association is released.

14.2 *Consumption of the submission port abstract-services*

This paragraph covers the invocation of the MessageSubmission, ProbeSubmission, and CancelDeferredDelivery abstract-operations, and the consumption of the SubmissionControl abstract-operation. The MS abstract-service consumption of the submission port abstract-services assumes that an abstract-association exists between the submission port supplier (the MTA) and the submission port consumer (the MS). The performance of the abstract-operations is in sequential order, no parallel processing takes place. Error cases are not described.

14.2.1 *Invocation of the MessageSubmission abstract-operation*

The initiation of a MessageSubmission abstract-association can be either from an auto-action within the MS or because the UA invoked a MessageSubmission abstract-operation to the MS. In order to submit the message to the MTA the MS performs the following steps:

1) If the MessageSubmission argument does not contain the forwarding-request extension (see § 6.6), it invokes a MessageSubmission abstract-operation, containing the message to be submitted and its associated parameters. For details, see § 8.2.1.1 of Recommendation X.411. Otherwise, checks to see that the entry is a delivered-message and incorporates information from one delivered-message entry in the stored-messages information-base, and then invokes the MessageSubmission abstract-operation with the new content. Forwarding of entries that are not delivered-messages is for further study.

   Note that although this forwarding-request is generic, it is not necessarily meaningful for all content-types. Where it is meaningful, the content-type of the referenced delivered-messages entry must be appropriate for incorporation into the content argument.

2) It gets a MessageSubmission result back when the MTA has accepted the submission. The MessageSubmission result contains among others information about identification of and submission-time for the submitted message. For details, see § 8.2.1.1.2 of Recommendation X.411.

3) If the MessageSubmission abstract-operation was triggered by a corresponding MessageSubmission abstract-operation to the MS from the UA, the result of the abstract-operation is passed back to the UA in the form of a MessageSubmission result issued by the MS. This behaviour guarantees that the message has actually been accepted by the MTA before the result is given back to the UA.

4) If the MTA has not accepted the message submission due to problems such as an invalid sequence number or inappropriate content-type, the MS will generate an error of InconsistentRequest. Note that all errors generated by the MTA are relayed through to the UA.

5) If a security-policy is in force, then to ensure that such a security-policy is not violated during message submission, the message-security-label is checked against the security-context by the MS. If the message submission is barred either by the security-policy or by temporary security restrictions, a security-error shall be indicated.

14.2.2 *Invocation of the ProbeSubmission abstract-operation*

A ProbeSubmission abstract-operation is initiated because the UA invoked a ProbeSubmission abstract-operation to the MS. In order to submit the probe to the MTA, the MS performs the following steps:

1) It invokes a ProbeSubmission abstract-operation, containing the probe to be submitted and its associated parameters. For details, see § 8.2.1.2.1 of Recommendation X.411.

2) It gets a ProbeSubmission result back when the MTA has accepted the submission. The result contains among others information about identification of and submission-time for the submitted probe. For details, see § 8.2.1.2.2 of Recommendation X.411.

3) The result of the abstract-operation is passed back to the UA in the form of a ProbeSubmission result issued by the MS. This behaviour guarantees that the probe has actually got accepted by the MTA before the result is given back to the UA.

4) If a security-policy is in force, then to ensure that such a security-policy is not violated during ProbeSubmission, the message-security-label of the probe is checked against the security-context by the MS. If the ProbeSubmission is barred either by the security-policy or by temporary security restrictions, a ProbeSubmission error is generated.

14.2.3 *Invocation of the CancelDeferredDelivery abstract-operation*

A CancelDeferredDelivery abstract-operation is initiated because the UA invoked a CancelDeferredDelivery abstract-operation to the MS. In order to send the cancel to the MTA, the MS performs the following steps:

1) It invokes a CancelDeferredDelivery abstract-operation, containing the cancel to be submitted and its associated parameters. For details, see § 8.2.1.3.1 of Recommendation X.411.
2) It gets a result back when the MTA has accepted the cancel. The result returned is empty as an indication of success.
3) The result of the abstract-operation is passed back to the UA in the form of a CancelDeferredDelivery result issued by the MS. This behaviour guarantees that the probe has actually got accepted (or not) by the MTA before the result is given back to the UA.

14.2.4 *Performance of the SubmissionControl abstract-operation*

If the MTA wants to temporarily stop the MS from submitting messages or probes, or to alter the maximum-content-length or lowest priority of messages from the MS, it invokes a SubmissionControl abstract-operation (for details, see § 8.2.1.4.1 of Recommendation X.411) to the MS. The MS reacts with the following steps:

1) It invokes a corresponding SubmissionControl abstract-operation from the MS to the UA.
2) It waits for the UA to send back a SubmissionControl result which contains information about whether messages or probes are waiting in the UA, due to the current restrictions. For details, see § 8.2.1.4.2 of Recommendation X.411.
3) The MS sends back a SubmissionControl result to the MTA, containing information from the UA.
4) When the MTS is able to accept any messages or probes again, it should invoke a new SubmissionControl abstract-operation to relax the restrictions. The effects of a SubmissionControl abstract-operation are cancelled when either a new SubmissionControl abstract-operation alters the restrictions or when the abstract-association is released. The MS then invokes a corresponding SubmissionControl abstract-operation to the UA and waits for the SubmissionControl result.

14.3 *Consumption of the administration port abstract-services*

This paragraph covers the performance of the register and ChangeCredentials abstract-operations. The consumption of the administration port abstract-services assumes that an abstract-association exists between the administration port supplier (the MTA) and the administration port consumer (the MS). The performance of the abstract-operations is in sequential order; no parallel processing takes place. Error cases are not described.

The MS use of the administration port is subject to the security-policy in force.

14.3.1 *Invocation of the register abstract-operation*

A register abstract-operation is initiated because the UA invoked a register abstract-operation to the MS. In order to send the registration to the MTA, the MS performs the following steps:

1) It invokes a register abstract-operation, containing the new data to be registered. For details, see § 8.4.1.1.1 of Recommendation X.411.
2) It gets a result back when the MTA has accepted the registration. The result returned is empty as an indication of success.
3) The scope of the permitted changes by the UA via the MS to the user-security-label arguments shall be confined to the security-policy in force.

14.3.2 *Invocation of the ChangeCredentials abstract-operation*

A ChangeCredentials abstract-operation is initiated because the UA invoked a ChangeCredentials abstract-operation to the MS. In order to relay the new credentials to the MTA from the UA, the MS performs the following steps:

1) It invokes a ChangeCredentials abstract-operation on the MTA, containing the new credentials to be registered. For details, see § 8.4.1.2.1 of Recommendation X.411.
2) It gets a ChangeCredentials result back when the MTA has accepted the change and stores the new credentials. The ChangeCredentials result or resultant error from the MTA is relayed to the UA and is empty as an indication of success.

14.3.3 *Performance of the ChangeCredentials abstract-operation*

When the MS receives a ChangeCredentials abstract-operation and its associated arguments from the MTA, it performs the following steps:

1) It establishes that the argument information is valid for a ChangeCredentials abstract-operation. For details, see § 8.4.1.2 of Recommendation X.411.
2) It checks if there is already an existing abstract-association between the MS and the UA. If an abstract-association between the MS and the UA does not exist, the MTA is informed by an error that change of credentials can not take place at present and no further steps are processed.
3) If the abstract-association between the MS and UA exists, the MS invokes a ChangeCredentials abstract-operation to the UA.
4) If the UA sends back an empty ChangeCredentials result, indicating success, the MS sends back a corresponding ChangeCredentials result indicating success to the MTA and stores the credentials. If the UA returns an error, this is relayed to the MTA to indicate that error. Note that the MS never sends back an indication of success to the MTA until it has received the corresponding result back from the UA first.

15 Supply of the message store abstract-service

This paragraph specifies how a MS supplies the MS abstract-service. Covered are its supply of the retrieval, indirect-submission, and administration ports.

15.1 *Supply of the retrieval port abstract-services*

This paragraph covers the supply of the summarize, list, fetch, delete, register-MS, and alert abstract-operations. The MS abstract-service supply of the retrieval port abstract-services assumes that an abstract-association exists between the retrieval port supplier (the MS) and the retrieval port consumer (the UA). The performance of the abstract-operations is in sequential order; no parallel processing takes place. Not all error cases are described.

15.1.1 *Performance of the summarize abstract-operation*

When the MS receives a summarize abstract-operation from the UA, it performs the following steps:

1) Establishes which information-base the summarize abstract-operation addresses.
2) Checks if there are any entries in the information-base. If it is empty, a summarize result with zero length is returned and no further steps are performed.
3) Checks that the supplied argument general-attributes and any content-specific attributes recognized by the MS are valid for a summarize abstract-operation. For details, see § 8.2.1.
4) Accumulates counts in accordance with the supplied argument general attributes and any content-specific attributes recognized by the MS.
5) Returns the summarize result to the UA. For details, see § 8.2.2.
6) If a security-policy is in force, then to ensure that such a security-policy is not violated during the summarize abstract-operation, the security classification of the security label is checked against the security-context by the MS. If a summarize is barred by the security-policy, the summarize abstract-operation shall be abandoned and a security error shall be indicated.

15.1.2 *Performance of the list abstract-operation*

When the MS receives a list abstract-operation from the UA, it performs the following steps:

1) Establishes which information-base the list abstract-operation addresses.
2) Checks that the supplied argument general-attributes and any content-specific attributes recognized by the MS are valid for a list abstract-operation. For details, see § 8.3.1.
3) Identifies zero or more entries as requested in the argument of the abstract-operation, up to any supplied limit. Child-entries to a parent-entry are excluded, unless explicitly selected in the argument.

4) If a set of requested general-attributes has been specified as arguments in the abstract-operation, these general-attributes are returned, if present, to the UA for each selected entry. If no request has been done, the default list abstract-operation values, as specified with a previous register-MS abstract-operation, are returned, if present. For more details, see § 8.3.2. The entry-status of each selected message is set to listed.

5) If a security-policy is in force, then to ensure that such a security-policy is not violated during the list abstract-operation, the message-security-label is checked against the security-context by the MS. If the list is barred either by the security-policy or by temporary security restrictions, the list abstract-operation shall be abandoned and a security error shall be indicated.

15.1.3 *Performance of the fetch abstract-operation*

When the MS receives a fetch abstract-operation from the UA, it performs the following steps:

1) Establishes which information-base the fetch abstract-operation addresses.

2) Checks that the supplied argument general-attributes and any content specific attributes recognized by the MS are valid for a fetch abstract-operation. For details, see § 8.4.1.

3) Identifies zero or more entries as requested in the argument of the abstract-operation, up to any supplied limit. Child-entries to a parent-entry are excluded, unless explicitly selected in the argument.

4) If a set of requested general-attributes have been specified as arguments in the abstract-operation, these general-attributes are returned, if present, to the UA for the first selected entry. If no request has been done, the default fetch abstract-operation values, as specified with a previous register-MS abstract-operation, are returned, if present. If several entries that match the search criteria are found, the sequence-numbers for the second and following entries are returned in increasing order. If there were more matching entries than in the specified limit, the next sequence number beyond the limit is also returned. For more details, see § 8.4.2.

5) If a security-policy is in force, then to ensure that such a security-policy is not violated during the fetch abstract-operation, the message-security-label is checked against the security-context by the MS. If the fetch abstract-operation is barred either by the security-policy or by temporary security restriction, the fetch abstract-operation shall be abandoned and a security error shall be indicated.

15.1.4 *Performance of the delete abstract-operation*

When the MS receives a delete abstract-operation from the UA, it performs the following steps:

1) Establishes which information-base the delete abstract-operation addresses.

2) Checks that the supplied arguments are valid for a delete abstract-operation. For details, see § 8.5.1.

3) Identifies the entry or list of entries requested in the argument of the abstract-operation.

4) If any of the entries has delete restrictions (see § 8.5), none of the deletions takes place. Otherwise all deletions are performed and an empty delete result returned to the UA as indication of success.

15.1.5 *Performance of the register-MS abstract-operation*

When the MS receives a register-MS abstract-operation from the UA, it performs the following steps:

1) Checks that the supplied arguments are valid for a register-MS abstract-operation. For details, see § 8.6.1.

2) Replaces any old parameters with the corresponding new ones. Auto-actions have effect on transactions, such as message-deliveries and report-deliveries, that occur after the initiation or deletion of auto-action requests; there is no processing of entries that already reside in the MS at that point in time.

3) Sends back an empty register-MS result to the UA to indicate that the abstract-operation has been performed successfully.

4) If a security-policy is in force, then the register-MS abstract-operation shall be subject to such a policy. Some security-policies may only permit user-security-labels to be changed if a secure link is employed. Other local means of changing the user-security-labels in a secure manner may be provided.

15.1.6 *Invocation of the alert abstract-operation*

The invocation of the alert abstract-operation is as a result of the consumption of the delivery port abstract-service (see § 14.1.1).

If the auto-alert auto-action is initiated by the UA, by an earlier register-MS abstract-operation, the MS abstract-service performs the following steps:

1) Checks if an abstract-association exists. If not, the MS will never establish an abstract-association, and no alert abstract-operation can be invoked.
2) If an abstract-association exists, the MS invokes the abstract-operation containing the relevant argument information (for details see § 8.7.1) and waits for a empty alert result to be returned by the UA as an indication of success.
3) If an abstract-association does not exist, there is a possibility to use a non-standardized protocol to inform the user. The alert signal in this case may be given on the user's terminal, but can alternatively be given on a telephone, a beeper or any other suitable terminal equipment associated with the user. The latter method can also be used in cases where the alert abstract-operation has not been implemented.
4) If a security-policy is in force, then to ensure that such a security-policy is not violated during the alert, the message-security-label is checked against the security-context by the MS. If the alert abstract-operation is barred either by the security-policy or by temporary security restrictions, the action taken shall be defined by the security-policy in force.

15.2 *Supply of the indirect-submission port abstract-services*

This paragraph covers the performance of the MessageSubmission, ProbeSubmission, and CancelDeferredDelivery abstract-operations, and the invocation of the SubmissionControl abstract-operation. The MS abstract-service supply of the indirect-submission port abstract-services assumes that an abstract-association exists between the indirect-submission port supplier (the MS) and the indirect-submission port consumer (the UA). The performance of the abstract-operations is in sequential order; no parallel processing takes place. Not all error cases are described.

15.2.1 *Performance of the MessageSubmission abstract-operation*

When the MS receives a MessageSubmission abstract-operation and its associated arguments from the UA, it performs the following steps:

1) It establishes that the argument information is valid for a MessageSubmission abstract-operation. For details, see § 8.2.1.1.1 of Recommendation X.411.
2) It checks the arguments to establish if the message content was supplied by the UA or if it has to be inserted by the MS (i.e., if the forwarding-request extension is present). In the latter case, if the entry is a delivered-message entry, the corresponding message is inserted and the MS-related arguments deleted. Forwarding of entries that are not delivered-messages is for further study.
3) It checks if there is aleady an existing abstract-association between the MS and the MTA. If not, the MS initiates such an abstract-association. If an abstract-association cannot be established, the UA is informed by an error that submission can not take place at present and no further steps are processed.
4) If the abstract-association between the MS and the MTA exists, the MS invokes a MessageSubmission abstract-operation to the MTA, after any modifications mentioned in step 2).
5) If the MTA sends back a MessageSubmission result (for details, see § 8.2.1.1.2 of Recommendation X.411) indicating success, the MS sends back a corresponding MessageSubmission result indicating success to the UA. Note that the MS never sends back an indication of success to the UA until it has received the corresponding result back from the MTA first. This is to insure a consistent service from a user point of view, viz., that a submission always means that the responsibility for the message has been taken over by the MTA when the result comes back.
6) The MS may either choose to terminate the abstract-association with the MTA after a certain period of inactivity, or when the UA terminates its corresponding abstract-association with the MS.

15.2.2 *Performance of the ProbeSubmission abstract-operation*

When the MS receives a ProbeSubmission abstract-operation and its associated arguments from the UA, it performs the following steps:

1) It establishes that the argument information is valid for a ProbeSubmission abstract-operation. For details, see § 8.2.1.2.1 of Recommendation X.411.

2) It checks if there is already an existing abstract-association between the MS and the MTA. If not, the MS initiates such an abstract-association. If an abstract-association cannot be established, the UA is informed by an error that submission can not take place at present and no further steps are processed.

3) If the abstract-association between the MS and the MTA exists, the MS invokes a ProbeSubmission abstract-operation to the MTA.

4) If the MTA sends back a ProbeSubmission result (for details, see § 8.2.1.2.2 of Recommendation X.411) indicating success, the MS sends back a corresponding ProbeSubmission result indicating success to the UA. Note that the MS never sends back an indication of success to the UA until it has received the corresponding result back from the MTA first. This is to ensure a consistent service from a user point of view, viz., that a submission always means that the responsibility for the probe has been taken over by the MTS when the result comes back.

5) The MS may either choose to terminate the abstract-association with the MTA after a certain period of inactivity, or when the UA terminates its corresponding abstract-association with the MS.

15.2.3 *Performance of the CancelDeferredDelivery abstract-operation*

When the MS receives a CancelDeferredDelivery abstract-operation and its associated arguments, it performs the following steps:

1) It establishes that the argument information is valid for a CancelDeferredDelivery abstract-operation. For details, see § 8.2.1.3.1 of Recommendation X.411.

2) It checks if there is already an existing abstract-association between the MS and the MTA. If not, the MS initiates such an abstract-association. If an abstract-association cannot be established, the UA is informed by an error that CancelDeferredDelivery can not take place at present and no further steps are processed.

3) If the abstract-association between the MS and the MTA exists, the MS invokes a CancelDeferredDelivery abstract-operation to the MTA.

4) If the MTA sends back a CancelDeferredDelivery result (for details, see § 8.2.1.3.2 of Recommendation X.411) indicating success, the MS sends back a corresponding CancelDeferredDelivery result indicating success to the UA. Note that the MS never sends back an indication of success to the UA until it has received the corresponding result back from the MTA first. This is to insure a consistent service from a user point of view, viz., that the responsibility for the cancel deferred delivery has been taken over by the MTS, when the result comes back.

5) The MS may either choose to terminate the abstract-association with the MTA after a certain period of inactivity, or when the UA terminates its corresponding abstract-association with the MS.

15.2.4 *Invocation of the SubmissionControl abstract-operation*

If the MS receives a SubmissionControl abstract-operation from the MTA, or if the MS for some internal reasons wants to temporarily stop the UA from submitting messages or probes, or to alter the maximum-length or lowest-priority of messages from the UA, the MS performs the following steps:

1) It invokes a SubmissionControl abstract-operation to the UA. For details, see § 8.2.1.4.1 of Recommendation X.411.

2) It waits for a SubmissionControl result (for details, see § 8.2.1.4.2 of Recommendation X.411) from the UA confirming the acceptance of the SubmissionControl abstract-operation.

3) If the SubmissionControl abstract-operation had been triggered by a corresponding abstract-operation from the MTA to the MS, the SubmissionControl result from the UA is passed on from the MS to the MTA and the MS waits for the SubmissionControl result to come back from the UA.

15.3 *Supply of the administration port abstract-services*

This paragraph covers the performance of the register and ChangeCredentials abstract-operations. The messages abstract-service supply of the administration port abstract-services assumes that an abstract-association exists between the indirect-submission port supplier (the MS) and the indirect-submission port consumer (the UA). The performance of the abstract-operations is in sequential order; no parallel processing takes place. Not all error cases are described.

15.3.1 *Performance of the register abstract-operation*

When the MS receives a register abstract-operation and its associated arguments from the UA, it performs the following steps:

1) It establishes that the argument information is valid for a register abstract-operation. For details, see § 8.4.1.1.1 of Recommendation X.411.
2) It checks if there is already an existing abstract-association between the MS and the MTA. If not, the MS initiates such an abstract-association. If an abstract-association cannot be established, the UA is informed by an error that register can not take place at present and no further steps are processed.
3) If the abstract-association between the MS and the MTA exists, the MS invokes a register abstract-operation to the MTA.
4) If the MTA sends back a register result (for details, see § 8.4.1.1.2 of Recommendation X.411) indicating success, the MS sends back a corresponding register result indicating success to the UA. Note that the MS never sends back an indication of success to the UA until it has received the corresponding result back from the MTA first. This is to ensure a consistent service from a user point of view, viz., that the responsibility for the register has been taken over by the MTS, when the result comes back.
5) The MS may either choose to terminate abstract-association with the MTA after a certain period of inactivity, or when the UA terminates its corresponding abstract-association with the MS.
6) The scope of permitted changes by the UA via the MS to the user-security-labels shall be confined by the security-policy in force. Some security-policies may only permit user-security-labels to be changed in this way if a secure link is employed. Other local means of changing user-security-labels in a secure manner may be provided.

15.3.2 *Invocation of the ChangeCredentials abstract-operation*

A ChangeCredentials abstract-operation is initiated because the MTA invoked a ChangeCredentials abstract-operation to the MS. In order to relay the new-credentials to the UA from the MTA, the MS performs the following steps:

1) It establishes that the argument information is valid for a ChangeCredentials abstract-operation. For details, see § 8.4.1.2 of Recommendation X.411. If the old credentials are incorrect and the new credentials are not acceptable, an error is returned and no further processing takes place.
2) It invokes a ChangeCredentials abstract-operation on the UA containing the new credentials to be registered. For details, see § 8.4.1.2 of Recommendation X.411.
3) It gets a ChangeCredentials result back when the UA has accepted the change and stores the new credentials. The ChangeCredentials result or resultant error from the UA is relayed to the MTA.

15.3.3 *Performance of the ChangeCredentials abstract-operation*

When the MS receives a ChangeCredentials abstract-operation and its associated arguments from the MTA, it performs the following steps:

1) It establishes that the argument information is valid for a ChangeCredentials abstract-operation. For details, see § 8.4.1.2 of Recommendation X.411.
2) It checks if there is already an existing abstract-association betwen the MS and the MTA. If not, the MS initiates such an abstract-association. If an abstract-association cannot be established, the UA is informed by an error that change of credentials can not take place at present and no further steps are processed.
3) If the abstract-association between the MS and MTA exists, the MS invokes a ChangeCredentials abstract-operation to the MTA.

4) If the MTA sends back an empty ChangeCredentials result, indicating success, the MS sends back a corresponding ChangeCredentials result indicating success to the UA and stores the credentials. If the MTA returns an error, this is relayed to the UA to indicate that error. Note that the MS never sends back an indication of success to the UA until it has received the corresponding result back from the MTA first.

5) The MS may either choose to terminate the abstract-association with the MTA after a certain period of inactivity, or when the UA terminates its corresponding abstract-association with the MS.

16 Ports realization

This paragraph describes how the retrieval, the submission and the administration ports of the MS abstract-service are provided. For a description of how the MTS abstract-service provides the delivery, the submission and the administration ports, refer to § 8 of Recommendation X.411.

16.1 *Retrieval port*

The retrieval port abstract-services are realized on a one-to-one basis between abstract-operations and real operations in the message retrieval service element (MRSE) which is documented in Recommendation X.419.

16.2 *Indirect-submission port*

The indirect-submission port abstract-services are realized on a one-to-one basis between abstract-operations and real operations in the message submission service element (MSSE) which is documented in Recommendation X.419.

16.3 *Administration port*

The administration port abstract-services are realized on a one-to-one basis between abstract-operations and real operations in the message administrtion service element (MASE) which is documented in Recommendation X.419.

ANNEX A

(to Recommendation X.413)

Formal assignment of object identifiers

This Annex is an integral part of this Recommendation.

All object identifiers this Recommendation assigns are formally assigned in the present Annex using ASN.1. The specified values are cited in the ASN.1 modules of subsequent annexes.

This Annex is definitive for all values except those for ASN.1 modules and for the whole subject matter of this Recommendation. The definitive assignments for the former occur in the modules themselves. The latter is fixed. Other references to the values assigned to modules appear in IMPORT clauses.

```
MSObjectIdentifiers
    { joint-iso-ccitt mhs-motis(6) ms(4) modules(0) object-identifiers(0) }
DEFINITIONS ::=
BEGIN


-- Prologue
-- Exports everything

IMPORTS
    ID, id-ms
        FROM MHSObjectIdentifiers { joint-iso-ccitt mhs-motis(6) arch(5) modules(0) object-identifiers(0) };
```

-- *Categories*

```
id-mod -- modules           -- ID ::= { id-ms 0 }
id-ot  -- objects           -- ID ::= { id-ms 1 }
id-pt  -- port types        -- ID ::= { id-ms 2 }
id-att -- attribute types   -- ID ::= { id-ms 3 }
id-act -- auto-action types -- ID ::= { id-ms 4 }
```

-- *Modules*

```
id-mod-object-identifiers   ID ::= { id-mod 0 } -- not definitive
id-mod-abstract-service     ID ::= { id-mod 1 } -- not definitive
id-mod-attribute-types      ID ::= { id-mod 2 } -- not definitive
id-mod-action-types         ID ::= { id-mod 3 } -- not definitive
id-mod-upper-bounds         ID ::= { id-mod 4 } -- not definitive
```

-- *Objects*

```
id-ot-ms                    ID ::= { id-ot 0 }
id-ot-ms-user               ID ::= { id-ot 1 }
```

-- *Port types*

```
id-pt-retrieval             ID ::= { id-pt 0 }
```

-- *Attribute types*

```
id-att-child-sequence-numbers                    ID ::= { id-att 0 }
id-att-content                                   ID ::= { id-att 1 }
id-att-content-confidentiality-algorithm-identifier ID ::= { id-att 2 }
id-att-content-correlator                        ID ::= { id-att 3 }
id-att-content-identifier                        ID ::= { id-att 4 }
id-att-content-integrity-check                   ID ::= { id-att 5 }
id-att-content-length                            ID ::= { id-att 6 }
id-att-content-returned                          ID ::= { id-att 7 }
id-att-content-type                              ID ::= { id-att 8 }
id-att-conversion-with-loss-prohibited           ID ::= { id-att 9 }
id-att-converted-EITs                            ID ::= { id-att 10 }
id-att-creation-time                             ID ::= { id-att 11 }
id-att-delivered-EITs                            ID ::= { id-att 12 }
id-att-delivery-flags                            ID ::= { id-att 13 }
id-att-dl-expansion-history                      ID ::= { id-att 14 }
id-att-entry-status                              ID ::= { id-att 15 }
id-att-entry-type                                ID ::= { id-att 16 }
id-att-intended-recipient-name                   ID ::= { id-att 17 }
id-att-message-delivery-envelope                 ID ::= { id-att 18 }
id-att-message-delivery-identifier               ID ::= { id-att 19 }
id-att-message-delivery-time                     ID ::= { id-att 20 }
id-att-message-origin-authentication-check       ID ::= { id-att 21 }
id-att-message-security-label                    ID ::= { id-att 22 }
id-att-message-submission-time                   ID ::= { id-att 23 }
id-att-message-token                             ID ::= { id-att 24 }
id-att-original-EITs                             ID ::= { id-att 25 }
id-att-originator-certificate                    ID ::= { id-att 26 }
id-att-originator-name                           ID ::= { id-att 27 }
id-att-other-recipient-names                     ID ::= { id-att 28 }
```

```
id-att-parent-sequence-number                    ID ::= { id-att 29 }
id-att-per-recipient-report-delivery-fields      ID ::= { id-att 30 }
id-att-priority                                  ID ::= { id-att 31 }
id-att-priority-of-delivery-request              ID ::= { id-att 32 }
id-att-redirection-history                       ID ::= { id-att 33 }
id-att-report-delivery-envelope,                 ID ::= { id-att 34 }
id-att-reporting-DL-name                         ID ::= { id-att 35 }
id-att-reporting-MTA-certificate                 ID ::= { id-att 36 }
id-att-report-origin-authentication-check        ID ::= { id-att 37 }
id-att-security-classification                   ID ::= { id-att 38 }
id-att-sequence-number                           ID ::= { id-att 39 }
id-att-subject-submission-identifier             ID ::= { id-att 40 }
id-att-this-recipient-name                       ID ::= { id-att 41 }


-- Auto-action types

id-act-auto-forward                              ID ::= { id-act 0 }
id-act-auto-alert                                ID ::= { id-act 1 }


END -- of MSObjectIdentifiers
```

ANNEX B

(to Recommendation X.413)

Formal definition of the message store abstract-service

This Annex is an integral part of this Recommendation.

This Annex, a supplement to Section 2, formally defines the message store abstract-service. It employs ASN.1 and the OBJECT, PORT, ABSTRACT-BIND, ABSTRACT-UNBIND, ABSTRACT-OPERATION, and ABSTRACT-ERROR macros of Recommendation X.407.

Note – The use of the ABSTRACT-BIND, ABSTRACT-UNBIND, ABSTRACT-OPERATION, and ABSTRACT-ERROR macros, which are derived from the BIND, UNBIND, OPERATION and ERROR macros of ROS, does not imply that the abstract-operations and abstract-errors are invoked and reported across the boundary between open-systems in every instance. However, frequently this will be done. Just how this is accomplished is the subject of Recommendation X.419.

```
MSAbstractService { joint-iso-ccitt mhs-motis(6) ms(4) modules(0) abstract-service(1) }
DEFINITIONS ::=

BEGIN


-- Prologue
-- Exports everything

IMPORTS


-- Abstract-services macros

ABSTRACT-BIND, ABSTRACT-ERROR, ABSTRACT-OPERATION, ABSTRACT-UNBIND, OBJECT, PORT
    FROM AbstractServiceNotation { joint-iso-ccitt mhs-motis(6) asdc(2) modules(0) notation(1) }


-- MS ports

administration, delivery, submission,
```

```
-- MTS macro

EXTENSION,


-- MTS abstract-service-data types

ContentLength, ContentType, Credentials, InitiatorCredentials, ORAddressAndOrDirectoryName,
ResponderCredentials, SecurityContext, SecurityError, SecurityLabel
        FROM MTSAbstractService { joint-iso-ccitt mhs-motis(6) mts(3) modules(0) mts-abstract-service(1) }


-- MS-objects

id-ot-ms, id-ot-ms-user, id-pt-retrieval
        FROM MSObjectIdentifiers { joint-iso-ccitt mhs-motis(6) ms(4) modules(0) object-identifiers(0) }

-- MS abstract-service upperbound

ub-attributes-supported, ub-attribute-values, ub-auto-actions, ub-auto-registrations,
ub-default-registrations, ub-error-reasons, ub-information-bases, ub-messages,
ub-nested-filters, ub-per-auto-action, ub-per-entry, ub-summaries
        FROM MSUpperBounds { joint-iso-ccitt mhs-motis(6) ms(4) modules(0) upper-bounds(4) }


-- MS abstract-service upperbound

ub-content-types, ub-encoded-information-types, ub-labels-and-redirections
        FROM MTSUpperBounds { joint-iso-ccitt mhs-motis(6) mts(3) modules(0) upper-bounds(3) };


-- MS abstract objects

MS OBJECT
        PORTS { retrieval[S],
                indirectSubmission[S],
                administration[S],
                delivery[C],
                submission[C],
                administration[C] }
        ::= id-ot-ms

msUser OBJECT
        PORTS { retrieval[C],
                indirectSubmission[C],
                administration[C] }
        ::= id-ot-ms-user


-- Port types

indirectSubmission PORT ::= submission

retrieval PORT
        CONSUMER INVOKES {
                Summarize,
                List,
                Fetch,
                Delete,
                Register-MS }
        SUPPLIER INVOKES {
                Alert }
        ::= id-pt-retrieval
```

```
-- Macros

AUTO-ACTION MACRO ::=
BEGIN
        TYPE NOTATION   ::= Registration
        VALUE NOTATION  ::= value (VALUE OBJECT IDENTIFIER)
        Registration    ::= "REGISTRATION PARAMETER IS" type

END


-- Common data-types related to the information model

InformationBase ::= INTEGER {
                stored-messages     (0),
                inlog               (1),
                outlog              (2) } (0. .ub-information-bases)

SequenceNumber ::= INTEGER (0. .ub-messages)

CreationTime ::= UTCTime

Attribute ::= SEQUENCE {
                type        AttributeType,
                values      SEQUENCE SIZE (1. .ub-attribute-values) OF ANY -- DEFINED BY TYPE -- }

AttributeType ::= OBJECT IDENTIFIER

AutoActionRegistration ::= SEQUENCE {
        type                        AutoActionType,
        registration-identifier     [0] INTEGER (1. .ub-per-auto-action) DEFAULT 1,
        registration-parameter      [1] ANY DEFINED BY type }

AutoActionType ::= OBJECT IDENTIFIER

EntryStatus ::= INTEGER {
        new         (0),
        listed      (1),
        processed   (2) }


-- Abstract-bind

MSBind ::= ABSTRACT-BIND
        TO { indirectSubmission[S], retrieval[S], administration[S] }
        BIND
                ARGUMENT        MSBindArgument
                RESULT          MSBindResult
                BIND-ERROR      MSBindError

MSUnbind ::= ABSTRACT-UNBIND
        FROM { indirectSubmission[S], retrieval[S], administration[S] }

MSBindArgument ::= SET {
        initiator-name              ORAddressAndOrDirectoryName
        initiator-credentials       [2] InitiatorCredentials,
        security-context            [3] IMPLICIT SecurityContext OPTIONAL,
        fetch-restrictions          [4] Restrictions OPTIONAL -- default is none --,
        ms-configuration-request    [5] BOOLEAN DEFAULT FALSE }

Restrictions ::= SET {
        allowed-content-types       [0] SET SIZE (1. .ub-content-types) OF OBJECT IDENTIFIER
                                        OPTIONAL
            -- default is no restriction --,
        allowed-EITs                [1] MS-EITs OPTIONAL -- default is no restriction --,
        maximum-content-length      [2] ContentLength OPTIONAL -- default is no restriction -- }
```

```
MS-EITs ::= SET SIZE (1..ub-encoded-information-types) OF MS-EIT

MS-EIT ::= OBJECT IDENTIFIER

MSBindResult ::= SET{
        responder-credentials           [2] ResponderCredentials,
        available-auto-actions          [3] SET SIZE (1..ub-auto-actions) OF AutoActionType OPTIONAL,
        available-attribute-types       [4] SET SIZE (1..ub-attributes-supported) OF AttributeType
                                            OPTIONAL,
        alert-indication                [5] BOOLEAN DEFAULT FALSE,
        content-types-supported         [6] SET SIZE (1..ub-content-types) OF OBJECT IDENTIFIER
                                            OPTIONAL}

MSBindError ::= ENUMERATED{
        authentication-error            (0),
        unacceptable-security-context   (1),
        unable-to-establish-association (2)}


-- Common data-types for abstract-operations

Range ::= CHOICE{
        sequence-number-range           [0] NumberRange,
        creation-time-range             [1] TimeRange}

NumberRange ::= SEQUENCE{
        from        [0] SequenceNumber OPTIONAL -- omitted means no lower bound --,
        to          [1] SequenceNumber OPTIONAL -- omitted means no upper bound --}

TimeRange ::= SEQUENCE{
        from        [0] CreationTime OPTIONAL -- omitted means no lower bound --,
        to          [1] CreationTime OPTIONAL -- omitted means no upper bound --}

Filter ::= CHOICE{
        item        [0] FilterItem,
        and         [1] SET SIZE (1..ub-nested-filters) OF Filter,
        or          [2] SET SIZE (1..ub-nested-filters) OF Filter,
        not         [3] Filter}

FilterItem ::= CHOICE{
        equality                [0] AttributeValueAssertion,
        substrings              [1] SEQUENCE{
            type                AttributeType,
            strings             SEQUENCE SIZE (1..ub-attribute-values) OF CHOICE{
                initial             [0] ANY -- DEFINED BY type --,
                any                 [1] ANY -- DEFINED BY type --,
                final               [2] ANY -- DEFINED BY type --}},
        greater-or-equal        [2] AttributeValueAssertion,
        less-or-equal           [3] AttributeValueAssertion,
        present                 [4] AttributeType,
        approximate-match       [5] AttributeValueAssertion}

AttributeValueAssertion ::= SEQUENCE{
        type        AttributeType,
        value       ANY DEFINED BY type}

Selector ::= SET{
        child-entries   [0] BOOLEAN DEFAULT FALSE,
        range           [1] Range OPTIONAL -- default is unbounded --,
        filter          [2] Filter OPTIONAL -- default is all entries within the specified range --,
        limit           [3] INTEGER (1..ub-messages) OPTIONAL,
        override        [4] OverrideRestrictions OPTIONAL -- default is that any fetch-restrictions in force do
                            apply --}

OverrideRestrictions ::= BIT STRING{
        overrideContentTypesRestriction     (0),
        overrideEITsRestriction             (1),
        overrideContentLengthRestriction    (2)} (SIZE (1..ub-information-bases))

EntryInformationSelection::= SET SIZE (0..ub-per-entry) OF AttributeSelection
```

```
AttributeSelection ::= SET{
        type            AttributeType,
        from            [0] INTEGER (1..ub-attribute-values) OPTIONAL -- used if type is multi valued --,
        count           [1] INTEGER (1..ub-attribute-values) OPTIONAL -- used if type is multi valued --}

EntryInformation ::= SEQUENCE{
        sequence-number         SequenceNumber,
        attributes              SET SIZE (1..ub-per-entry) OF Attribute OPTIONAL}


-- Forwarding-request parameter for indirect-submission

forwarding-request EXTENSION
        SequenceNumber
        CRITICAL FOR SUBMISSION
        ::= 36


-- Abstract-operations

Summarize ::= ABSTRACT-OPERATION
        ARGUMENT        SummarizeArgument
        RESULT          SummarizeResult
        ERRORS{
                AttributeError,
                InvalidParametersError,
                RangeError,
                SecurityError,
                SequenceNumberError,
                ServiceError}

SummarizeArgument ::= SET{
        information-base-type   [0] InformationBase DEFAULT stored-messages,
        selector                [1] Selector,
        summary-requests        [2] SEQUENCE SIZE (1..ub-summaries) OF AttributeType OPTIONAL
                -- absent if no summaries are requested --}

SummarizeResult ::= SET{
        next            [0] SequenceNumber OPTIONAL,
        count           [1] INTEGER (0..ub-messages) -- of the entries selected --,
        span            [2] Span OPTIONAL -- of the entries selected, omitted if count is zero --,
        summaries       [3] SEQUENCE SIZE (1..ub-summaries) OF Summary OPTIONAL}

Span ::= SEQUENCE{
        lowest          [0] SequenceNumber,
        highest         [1] SequenceNumber}

Summary ::= SET{
        absent          [0] INTEGER (1..ub-messages) OPTIONAL -- count of entries where the attribute is
                            absent --,
        present         [1] SET SIZE (1..ub-attribute-values) OF -- one for each attribute value present --
                                SEQUENCE{
                                    type        AttributeType,
                                    value       ANY DEFINED BY type,
                                    count       INTEGER (1..ub-messages)} OPTIONAL}

List ::= ABSTRACT-OPERATION
                ARGUMENT        ListArgument
                RESULT          ListResult
                ERRORS{
                    AttributeError,
                    InvalidParametersError,
                    RangeError,
                    SecurityError,
                    SequenceNumberError,
                    ServiceError}
```

```
ListArgument ::= SET{
        information-base-type       [0] InformationBase DEFAULT stored-messages,
        selector                    [1] Selector,
        requested-attributes        [3] EntryInformationSelection OPTIONAL}

ListResult ::= SET{
        next            [0] SequenceNumber OPTIONAL,
        requested       [1] SEQUENCE SIZE (1..ub-messages) OF EntryInformation OPTIONAL -- omitted if
                            none found --}

--

Fetch ::= ABSTRACT-OPERATION
                ARGUMENT            FetchArgument
                RESULT              FetchResult
                ERRORS{
                    AttributeError,
                    FetchRestrictionError,
                    InvalidParametersError,
                    RangeError,
                    SecurityError,
                    SequenceNumberError,
                    ServiceError}

FetchArgument ::= SET{
        information-base-type       [0] InformationBase DEFAULT stored-messages,
        item                            CHOICE{
                search                          [1] Selector,
                precise                         [2] SequenceNumber},
        requested-attributes        [3] EntryInformationSelection OPTIONAL}

FetchResult ::= SET{
        entry-information           [0] EntryInformation OPTIONAL -- if an entry was selected --,
        list                        [1] SEQUENCE SIZE (1..ub-messages) OF SequenceNumber OPTIONAL,
        next                        [2] SequenceNumber OPTIONAL}

--

Delete ::= ABSTRACT-OPERATION
                ARGUMENT            DeleteArgument
                RESULT              DeleteResult
                ERRORS{
                    DeleteError,
                    InvalidParametersError,
                    RangeError,
                    SecurityError,
                    SequenceNumberError,
                    ServiceError}

DeleteArgument ::= SET{
        information-base-type       [0] InformationBaseDEFAULT stored-messages,
        items                           CHOICE{
                selector                        [1] Selector
                sequence-numbers                [2] SET SIZE (1..ub-messages) OF SequenceNumber}}

DeleteResult ::= NULL

--

Register-MS ::= ABSTRACT-OPERATION
                ARGUMENT            Register-MSArgument
                RESULT              Register-MSResult
                ERRORS{
                    AttributeError,
                    AutoActionRequestError,
                    InvalidParametersError,
                    SecurityError,
                    ServiceError}
```

```
Register-MSArgument ::= SET{
        auto-action-registrations       [0] SET SIZE (1..ub-auto-registrations) OF AutoActionRegistration
                                            OPTIONAL,
        auto-action-deregistrations     [1] SET SIZE (1..ub-auto-registrations) OF AutoActionDeregistration
                                            OPTIONAL,
        list-attribute-defaults         [2] SET SIZE (1..ub-default-registrations) OF AttributeType OPTIONAL,
        fetch-attribute-defaults        [3] SET SIZE (1..ub-default-registrations) OF AttributeType OPTIONAL,
        change-credentials              [4] SEQUENCE{
                old-credentials                     [0] IMPLICIT Credentials,
                new-credentials                     [1] IMPLICIT Credentials} OPTIONAL
                    -- same CHOICE as for old-credentials --,
        user-security-labels            [5] SET SIZE (1..ub-labels-and-redirections) OF SecurityLabel OPTIONAL}

AutoActionDeregistration ::= AutoActionRegistration
                                (WITH COMPONENTS {..., registration-parameter ABSENT})

Register-MSResult ::= NULL

--

Alert ::= ABSTRACT-OPERATION
                ARGUMENT        AlertArgument
                RESULT          AlertResult
                ERRORS{
                    SecurityError}

AlertArgument ::= SET{
        alert-registration-identifier   [0] INTEGER (1..ub-auto-actions),
        new-entry                       [2] EntryInformation OPTIONAL}

AlertResult ::= NULL


-- Abstract-errors

AttributeError ::= ABSTRACT-ERROR
        PARAMETER SET{
                problems [0]        SET SIZE (1..ub-per-entry) OF SET{
                    problem         [0] AttributeProblem,
                    type            [1] AttributeType,
                    value           [2] ANY DEFINED BY type OPTIONAL}}

AttributeProblem ::= INTEGER{
        invalid-attribute-value         (0),
        unavailable-attribute-type      (1),
        inappropriate-matching          (2),
        attribute-type-not-subscribed   (3),
        inappropriate-for-operation     (4)} (0..ub-error-reasons)

--

AutoActionRequestError ::= ABSTRACT-ERROR
        PARAMETER SET{
                problems [0]        SET SIZE (1..ub-auto-registrations) OF SET{
                    problem         [0] AutoActionRequestProblem,
                    type            [1] AutoActionType}}

AutoActionRequestProblem ::= INTEGER{
        unavailable-auto-action-type    (0),
        auto-action-type-not-subscribed (1)} (0..ub-error-reasons)

--
```

```
DeleteError ::= ABSTRACT-ERROR
        PARAMETER SET{
                problems [0]            SET SIZE (1..ub-messages) OF SET{
                    problem             [0] DeleteProblem,
                    sequence-number     [1] SequenceNumber}}

DeleteProblem ::= INTEGER{
        child-entry-specified           (0),
        delete-restriction-problem      (1)} (0..ub-error-reasons)

--

FetchRestrictionError ::= ABSTRACT-ERROR
        PARAMETER SET{
                problems [0]            SET SIZE (1..ub-default-registrations) OF SET{
                    problem             [3] FetchRestrictionProblem,
                    restriction         CHOICE{
                        content-type        [1] ContentType,
                        eit                 [2] MS-EITs,
                        content-length      [3] ContentLength}}}

FetchRestrictionProblem ::= INTEGER{
        content-type-problem    (1),
        eit-problem             (2),
        content-length-problem  (3)} (0..ub-error-reasons)

--

InvalidParametersError ::= ABSTRACT-ERROR
        PARAMETER  NULL

--

RangeError ::= ABSTRACT-ERROR

        PARAMETER       SET{
                problem     [0] RangeProblem}

RangeProblem ::= INTEGER{
        reversed    (0)} (0..ub-error-reasons)

--

SequenceNumberError ::= ABSTRACT-ERROR
        PARAMETER SET{
                problems [1]            SET SIZE (1..ub-messages) OF SET{
                    problem             [0] SequenceNumberProblem,
                    sequence-number     [1] SequenceNumber}}

SequenceNumberProblem ::= INTEGER{
        no-such-entry    (0)} (0..ub-error-reasons)

--

ServiceError ::= ABSTRACT-ERROR
        PARAMETER       SET{
                problem     [0] ServiceProblem}

ServiceProblem ::= INTEGER{
        busy                    (0),
        unavailable             (1),
        unwilling-to-perform    (2)} (0..ub-error-reasons)


END  -- of MSAbstractService
```

ANNEX C

(to Recommendation X.413)

Formal definition of general-attribute-types

This Annex is an integral part of this Recommendation.

This Annex, a supplement to section 3, formally defines the general-attribute-types applicable to all forms of message handling, rather than just one. It employs ASN.1 and the ATTRIBUTE macro.

```
MSGeneralAttributeTypes { joint-iso-ccitt mhs-motis(6) ms(4) modules(0) general-attribute-types(2) }
DEFINITIONS ::=

BEGIN


-- Prologue
-- Exports everything


IMPORTS

    -- Identificadores de objeto de tipos de atributos generales
    id-att-child-sequence-numbers, id-att-content, id-att-content-confidentiality-algorithm-identifier,
    id-att-content-correlator, id-att-content-identifier, id-att-content-integrity-check, id-att-content-length,
    id-att-content-returned, id-att-content-type, id-att-conversion-with-loss-prohibited, id-att-converted-EITs,
    id-att-creation-time, id-att-delivered-EITs, id-att-delivery-flags, id-att-dl-expansion-history,
    id-att-entry-status, id-att-entry-type, id-intended-recipient-name, id-att-message-delivery-envelope,
    id-att-message-delivery-identifier, id-att-message-delivery-time, id-att-message-origin-authentication-check,
    id-att-message-security-label, id-att-message-submission-time, id-att-message-token, id-att-original-EITs,
    id-att-originator-certificate, id-att-originator-name, id-att-other-recipient-names,
    id-att-parent-sequence-number, id-att-priority, id-att-proof-of-delivery-request, id-att-redirection-history,
    id-att-report-delivery-envelope, id-att-reporting-DL-name, id-att-reporting-MTA-certificate,
    id-att-report-origin-authentication-check, id-att-sequence-number, id-att-subject-submission-identifier,
    id-att-this-recipient-name
        FROM MSObjectIdentifiers { joint-iso-ccitt mhs-motis(6) ms(4) modules(0) object-identifiers(0) }

    -- Attribute macros
    ATTRIBUTE, ATTRIBUTE-SYNTAX
        FROM InformationFramework { joint-iso-ccitt ds(5) modules(1) informationFramework(1) }

    -- MS abstract-service data-types
    CreationTime, EntryStatus, MS-EIT, SequenceNumber
        FROM MSAbstractService { joint-iso-ccitt mhs-motis(6) ms(4) modules(0) abstract-service(1) }

    -- Authentication-service data-types
    AlgorithmIdentifier
        FROM AuthenticationFramework { joint-iso-ccitt ds(5) modules(1) authentication-framework(7) }

    -- MTS abstract-service data-types
    Content, ContentCorrelator, ContentIdentifier, ContentIntegrityCheck, ContentLength,
    ConversionWithLossProhibited, DeliveryFlags, DLExpansionHistory, MessageDeliveryEnvelope,
    MessageDeliveryIdentifier, MessageDeliveryTime, MessageOriginAuthenticationCheck,
    MessageSecurityLabel, MessageSubmissionTime, MessageToken, OriginatorCertificate, ORName,
    PerRecipientReportDeliveryFields, Priority, ProofOfDeliveryRequest, RedirectionHistory,
    ReportDeliveryEnvelope, ReportingDLName, ReportingMTACertificate,
    ReportOriginAuthenticationCheck, SecurityClassification, subjectSubmissionIdentifier
        FROM MTSAbstractService { joint-iso-ccitt mhs-motis(6) mts(3) modules(0)
        mts-abstract-service(1) }

    -- MS abstract-service upperbound
    ub-entry-types
        FROM MSUpperBounds { joint-iso-ccitt mhs-motis(6) ms(4) modules(0) upper-bounds(4) };
```

-- *Attribute-types*

```
ms-child-sequence-numbers ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX SequenceNumber
        MULTI VALUE
        ::= id-att-child-sequence-numbers


ms-content ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX Content
        SINGLE VALUE
        ::= id-att-content


mt-content-confidentiality-algorithm-identifier ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX AlgorithmIdentifier
        SINGLE VALUE
        ::= id-att-content-confidentiality-algorithm-identifier


mt-content-correlator ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ContentCorrelator
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-content-correlator


mt-content-identifier ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ContentIdentifier
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-content-identifier


mt-content-integrity-check ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ContentIntegrityCheck
        SINGLE VALUE
        ::= id-att-content-integrity-check


ms-content-length ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ContentLength
        MATCHES FOR ORDERING
        SINGLE VALUE
        ::= id-att-content-length


ms-content-returned ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX BOOLEAN
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-content-returned


mt-content-type ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX OBJECT IDENTIFIER
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-content-type


mt-conversion-with-loss-prohibited ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ConversionWithLossProhibited
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-conversion-with-loss-prohibited
```

```
ms-converted-EITs ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MS-EIT
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-att-converted-EITs

ms-creation-time ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX CreationTime
        MATCHES FOR EQUALITY ORDERING
        SINGLE VALUE
        ::= id-att-creation-time

ms-delivered-EITs ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MS-EIT
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-att-delivered-EITs

mt-delivery-flags ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX DeliveryFlags
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-delivery-flags

mt-dl-expansion-history ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX DLExpansionHistory
        MULTI VALUE
        ::= id-att-dl-expansion-history

ms-entry-status ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX EntryStatus
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-entry-status

ms-entry-type ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX EntryType
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-entry-type

EntryType ::= INTEGER {
        delivered-message       (0),
        delivered-report        (1),
        returned-content        (2) (0. .ub-entry-types) }

mt-intended-recipient-name ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ORName
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-intended-recipient-name

mt-message-delivery-envelope ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MessageDeliveryEnvelope
        SINGLE VALUE
        ::= id-att-message-delivery-envelope

mt-message-delivery-identifier ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MessageDeliveryIdentifier
        SINGLE VALUE
        ::= id-att-message-delivery-identifier

mt-message-delivery-time ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MessageDeliveryTime
        MATCHES FOR EQUALITY ORDERING
        SINGLE VALUE
        ::= id-att-message-delivery-time
```

```
mt-message-origin-authentication-check ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MessageOriginAuthenticationCheck
        SINGLE VALUE
        ::= id-att-message-origin-authentication-check

mt-message-security-label ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MessageSecurityLabel
        SINGLE VALUE
        ::= id-att-message-security-label

mt-message-submission-time ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MessageSubmissionTime
        MATCHES FOR EQUALITY ORDERING
        SINGLE VALUE
        ::= id-att-message-submission-time

mt-message-token ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MessageToken
        SINGLE VALUE
        ::= id-att-message-token

ms-original-EITs ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MS-EIT
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-att-original-EITs

mt-originator-certificate ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX OriginatorCertificate
        SINGLE VALUE
        ::= id-att-originator-certificate

mt-originator-name ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ORName
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-originator-name

mt-other-recipient-names ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ORName
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-att-other-recipient-names

ms-parent-sequence-number ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX SequenceNumber
        MATCHES FOR EQUALITY ORDERING
        SINGLE VALUE
        ::= id-att-parent-sequence-number

mt-per-recipient-report-delivery-fields ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX PerRecipientReportDeliveryFields
        MULTI VALUE
        ::= id-att-per-recipient-report-delivery-fields

mt-priority ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX Priority
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-priority

mt-proof-of-delivery-request ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ProofOfDeliveryRequest
        SINGLE VALUE
        ::= id-att-proof-of-delivery-request

mt-redirection-history ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX RedirectionHistory
        MULTI VALUE
        ::= id-att-redirection-history
```

```
mt-report-delivery-envelope ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ReportDeliveryEnvelope
        SINGLE VALUE
        ::= id-att-report-delivery-envelope

mt-reporting-DL-name ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ReportingDLName
        SINGLE VALUE
        ::= id-att-reporting-DL-name

mt-reporting-MTA-certificate ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ReportingMTACertificate
        SINGLE VALUE
        ::= id-att-reporting-MTA-certificate

mt-report-origin-authentication-check ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ReportOriginAuthenticationCheck
        SINGLE VALUE
        ::= id-att-report-origin-authentication-check

mt-security-classification ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX SecurityClassification
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-security-classification

ms-sequence-number ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX SequenceNumber
        MATCHES FOR EQUALITY ORDERING
        SINGLE VALUE
        ::= id-att-sequence-number

mt-subject-submission-identifier ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX SubjectSubmissionIdentifier
        SINGLE VALUE
        ::= id-att-subject-submission-identifier

mt-this-recipient-name ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ORName
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-att-this-recipient-name


END  -- of MSGeneralAttributeTypes
```

ANNEX D

(to Recommendation X.413)

Formal definition of general-auto-action-types

This Annex is an integral part of this Recommendation.

This Annex, a supplement to Section 3, formally defines the general-auto-action-types applicable to all forms of message handling, rather than just one. It employs ASN.1 and the AUTO-ACTION macro.

```
MSGeneralAutoActionTypes {joint-iso-ccitt mhs-motis(6) ms(4) modules(0) general-auto-action-types(3)}
DEFINITION ::=

BEGIN

-- Prologue


EXPORTS

        -- General-auto-action-types
        auto-forward, auto-alert;
```

```
IMPORTS
        -- General-auto-action-type object identifiers
        id-act-auto-forward, id-act-auto-alert
                FROM MSObjectIdentifiers {joint-iso-ccitt mhs-motis(6) ms(4) modules(0) object-identifiers(0)}

        -- Auto-action macro
        AUTO-ACTION,

        -- MS abstract-service data-types
        Content, Filter, EntryInformationSelection
                FROM MSAbstractService {joint-iso-ccitt mhs-motis(6) ms(4) modules(0) abstract-service(1)}

        -- MTS abstract-service data-types
        ContentIdentifier, DeferredDeliveryTime, ExplicitConversion, OriginatorName, OriginatorReportRequest,
        PerMessageIndicators, PerMessageSubmissionExtensions, PerRecipientMessageSubmissionExtensions,
        Priority, RecipientName
                FROM MTSAbstractService {joint-iso-ccitt mhs-motis(6) mts(3) modules(0) mts-abstract-
                service(1)}

        -- MS abstract-service upperbound
        ub-alert-addresses
                FROM MSUpperBounds {joint-iso-ccitt mhs-motis(6) ms(4) modules(0) upper-bounds(4)};

        -- Action-types
        auto-forward AUTO-ACTION
                REGISTRATION PARAMETER IS AutoForwardRegistrationParameter
                ::= id-act-auto-forward

AutoForwardRegistrationParameter ::= SET{
                filter                          [0] Filter OPTIONAL,
                auto-forward-arguments          [1] AutoForwardArguments,
                delete-after-auto-forwarding    [2] BOOLEAN DEFAULT TRUE,
                other-parameters                [3] OCTET STRING OPTIONAL}

AutoForwardArguments ::= SET{
                COMPONENTS OF PerMessageAutoForwardFields,
                per-recipient-fields    [1] IMPLICIT SEQUENCE SIZE (1..ub-recipients) OF PerRecipient-
                                            AutoForwardFields}

PerMessageAutoForwardFields ::= SET{
                originator-name         OriginatorName,
                content-identifier      ContentIdentifier OPTIONAL,
                priority                Priority DEFAULT normal,
                per-message-indicators  PerMessageIndicators DEFAULT {},
                deferred-delivery-time  [0] IMPLICIT DeferredDeliveryTime OPTIONAL,
                extensions              [2] IMPLICIT PerMessageSubmissionExtensions DEFAULT {}}

PerRecipientAutoForwardFields ::= SET{
                recipient-name          RecipientName,
                originator-report-request   [0] IMPLICIT OriginatorReportRequest,
                explicit-conversion     [1] IMPLICIT ExplicitConversion OPTIONAL,
                extensions              [2] IMPLICIT PerRecipientMessageSubmissionExtensions
                                            DEFAULT {}}

auto-alert AUTO-ACTION
                REGISTRATION PARAMETER IS AutoAlertRegistrationParameter
                ::= id-act-auto-alert

AutoAlertRegistrationParameter ::= SET{
                filter                  [0] Filter OPTIONAL,
                alert-addresses         [1] SEQUENCE SIZE (1..ub-alert-addresses) OF AlertAddress
                                            OPTIONAL,
                requested-attributes    [2] EntryInformationSelection OPTIONAL}

AlertAddress ::= SEQUENCE{
                address         EXTERNAL,
                alert-qualifier OCTET STRING OPTIONAL}


END -- of MSGeneralAutoActionTypes
```

ANNEX E

(to Recommendation X.413)

Formal definition of MS parameter upper bounds

This Annex is an integral part of this Recommendation.

This Annex defines for reference purpose the upper bounds of various variable length data types whose abstract syntaxes are defined in ASN.1 modules in the body of this Recommendation.

```
MSUpperBounds { joint-iso-ccitt mhs-motis(6) ms(4) modules(0) upper-bounds(4) }
DEFINITIONS IMPLICIT TAGS ::=
BEGIN

-- Prologue
-- Exports everything
IMPORTS -- nothing --;

-- Upper bounds
ub-alert-addresses          INTEGER ::= 16
ub-attribute-values         INTEGER ::= 32767       -- (2^15 - 1) the largest integer representable in 16 bits --
ub-attributes-supported     INTEGER ::= 1024
ub-auto-actions             INTEGER ::= 16
ub-auto-registrations       INTEGER ::= 1024
ub-default-registrations    INTEGER ::= 1024
ub-entry-types              INTEGER ::= 16
ub-error-reasons            INTEGER ::= 16
ub-information-bases        INTEGER ::= 16
ub-messages                 INTEGER ::= 2147483647  -- (2^31 - 1) the largest integer representable in 32 bits --
ub-nested-filters           INTEGER ::= 32
ub-per-auto-action          INTEGER ::= 32767       -- (2^15 - 1) the largest integer representable in 16 bits --
ub-per-entry                INTEGER ::= 1024
ub-summaries                INTEGER ::= 16

END -- of MSUpperBounds
```

ANNEX F

Example of the summarize abstract-operation

This Annex is not part of this Recommendation.

This Annex contains an example of the use of the summarize abstract-operation.

F.1 *The entries in the example MS*

Consider an MS containing the following entries, one entry per line. The columns show the values of the indicated attribute-types. A "–" indicates that the attribute is absent from the entry.

TABLE F-1/X.413

Stored-messages in the example

Sequence number	Entry-type	Entry-status	Priority
3	message	listed	urgent
5	message	listed	low
8	report	listed	–
10	message	listed	normal
15	report	new	–
18	message	new	normal
20	message	new	urgent
22	message	new	normal
23	message	new	normal

Note – Even if the priority in a message-delivery-envelope of a message is omitted and defaulted to "normal", the corresponding attribute is present with its value set to the default.

F.2 *A example of a request for summary*

Suppose the requirement is to summarize all the "new" entries by priority. The required result is the following list of counts. The numbers in parenthesis are sequence-numbers of the messages contributing to that count. See Table F-2/X.413.

TABLE F-2/X.413

Expected result from the list-summary

Priority	Count
–	1 (15)
urgent	1 (20)
normal	3 (18,22,23)
low	0

The components of the summarize-argument should be set as follows:

selector:

filter: Entry-status = new

summary-requests: attribute type = Priority

The components of the summarize-result might be as follows:

count: 5
span:
lowest: 15
highest: 23

summaries:
{absent: 1
present: {value = normal, count = 3}
{value = urgent, count = 1}}

ANNEX G

Differences between the CCITT Recommendation X.413 text and ISO/IEC 10021-5 text

This Annex is not part of this Recommendation.

There are only two known differences between Recommendation X.413 in CCITT and the MOTIS 10021-5 in ISO/IEC.

1) The CCITT text contains a restriction in § 7.1 that only one abstract-association may exist at any time between the MS and the MS-user. This restriction is *not* included in the ISO/IEC text.

2) Those parts of the ASN.1 notation which express upper bounds and are documented in Annex E, are not considered to be part of the MOTIS standard, but are a formal part of Recommendation X.413.

   In ISO, this level of functionality is the responsibility of the Special Group on Functional Standardization, which publishes Internationally Standardized Profiles (ISPs), containing e.g. upper bounds for protocol elements.

Recommendation X.419

MESSAGE HANDLING SYSTEMS: PROTOCOL SPECIFICATIONS [1]

(Melbourne, 1988)

The establishment in various countries of telematic services and computer-based store-and-forward message services in association with public data networks creates a need to produce standards to facilitate international message exchange between subscribers to such services.

The CCITT,

considering

(a) the need for message handling systems;

(b) the need to transfer and store messages of different types;

(c) that Recommendation X.200 defines the reference model of open systems interconnection for CCITT applications;

(d) that Recommendations X.208, X.217, X.218 and X.219 provide the foundation for CCITT applications;

(e) that the X.500-series Recommendations define directory systems;

(f) that message handling systems are defined in a series of Recommendations: X.400, X.402, X.403, X.407, X.408, X.411, X.413 and X.419; and

(g) that interpersonal messaging is defined in Recommendations X.420 and T.330;

unanimously declares

(1) that the protocol for accessing the message transfer system (the MTS access protocol – P3) is defined in Section 2;

(2) that the protocol for accessing a message store (the MS access protocol – P7) is also defined in Section 2;

(3) that the protocol used between message transfer agents (MTAs) to provide for the distributed operation of the message transfer system (the MTS transfer protocol – P1) is defined in Section 3.

TABLE OF CONTENTS

[1] Recommendation X.419 and ISO 10021-6 [Information Processing Systems – Text Communication – MOTIS – Protocol Specifications] were developed in close collaboration and are technically aligned, except for the differences noted in Annex D.

0 Introduction

This Recommendation is one of a set of Recommendations defining message handling in a distributed open systems environment.

Message handling provides for the exchange of messages between users on a store-and-forward basis. A message submitted by one user (the *originator*) is transferred through the message transfer system (MTS) and delivered to one or more other users (the *recipients*). A user may interact directly with the MTS, or indirectly via a message store (MS).

The MTS comprises a number of message-transfer-agents (MTAs), which transfer messages and deliver them to their intended recipients.

This Recommendation was developed jointly by CCITT and ISO. The equivalent ISO document is ISO 10021-6.

1 Scope

This Recommendation specifies the MTS access protocol (P3) used between a remote user-agent and the MTS to provide access to the MTS abstract service defined in Recommendation X.411.

This Recommendation also specifies the MS access protocol (P7) used between a remote user-agent and a message-store (MS) to provide access to the MS abstract service defined in Recommendation X.413.

This Recommendation also specifies the MTS transfer protocol (P1) used between MTASs to provide the distributed operation of the MTS as defined in Recommendation X.411.

Recommendation X.402 identifies the other Recommendations which define other aspects of message handling systems.

Section 2 of this Recommendation specifies the MHS access protocols (P3 and P7). Paragraph 6 provides an overview of the MHS access protocols. Paragraph 7 defines the abstract-syntax of the MTS access protocol (P3). Paragraph 8 defines the abstract-syntax of the MS access protocol (P7). Paragraph 9 defines the mapping of the MHS access protocols onto used services. Paragraph 10 specifies conformance requirements for systems implementing the MHS access protocols.

Section 3 of this Recommendation specifies the MTS transfer protocol (P1). Paragraph 11 provides an overview of the MTS transfer protocol (P1). Paragraph 12 defines the abstract-syntax of the MTS transfer protocol (P1). Paragraph 13 defines the mapping of the MTS transfer protocol (P1) onto used services. Paragraph 14 specifies conformance requirements for systems implementing the MTS transfer protocol (P1).

Annex A provides a reference definition of the MHS protocol object identifiers cited in the ASN.1 modules in the body of this Recommendation.

Annex B describes protocol rules for interworking with implementations of the Recommendation X.411 (1984) using the MTS Transfer Protocol (P1).

Annex C identifies the differences between the Recommendation X.411 (1984) and this Recommendation.

Annex D identifies the technical differences between the ISO and CCITT versions of CCITT Recommendations X.419 and ISO 10021-6.

2 References

References are listed in Recommendation X.402.

3 Definitions

Definitions are given in Recommendation X.402.

4 Abbreviations

Abbreviations are listed in Recommendation X.402.

5 Conventions

This Recommendation uses the descriptive conventions described below.

5.1 *Terms*

Throughout this Recommendation the words of defined terms, and the names and values of service parameters and protocol fields, unless they are proper names, begin with a lower-case letter and are linked by a hyphen thus: defined-terms. Proper names begin with an upper-case letter and are not linked by a hyphen thus: Proper Name.

5.2 *Abstract syntax definitions*

This Recommendation defines the abstract-syntax of the MHS protocols using the abstract syntax notation (ASN.1) defined in Recommendation X.208 and the remote operations notation defined in Recommendation X.219.

6 Overview of the MHS access protocols

6.1 *MHS access protocol model*

Paragraph 6 of Recommendation X.411 describes an abstract model of the message transfer system (MTS), and the MTS abstract service which it provides to its MTS-users.

Paragraph 6 of Recommendation X.413 describes an abstract model of a message store (MS), and the MTS abstract service which it provides to its MS-users.

This paragraph describes how the MTS abstract service and the MS abstract service are supported by instances of OSI communication when an abstract-service user and an abstract-service provider are realized as application-processes located in different open systems.

In the OSI environment, communication between application-processes is represented in terms of communication between a pair of application-entities (AEs) using the presentation-service. The functionality of an application-entity is factored into a set of one or more application-service-elements (ASEs). The interaction between AEs is described in terms of their use of the services provided by the ASEs.

Access to the MTS abstract service is supported by three application-service-elements, each supporting a type of port paired between an MTS-user and the MTS in the abstract model. The message submission service element (MSSE) supports the services of the submission-port; the message delivery service element (MDSE) supports the services of the delivery-port; and the message administration service element (MASE) supports the services of the administration-port. The MSSE, MDSE and MASE are asymmetric-ASEs; that is, the MTS-user ASEs act as the consumer, and the MTS ASEs act as the supplier, of the MTS abstract service.

Similarly, access to the MS abstract service is supported by three application-service-elements: the message submission service element (MSSE) supports the indirect-submission-port; the message retrieval service element (MRSE) supports the services of the retrieval-port; and the message administration service element (MASE) supports the services of the administration-port. The MS-user ASEs act as the consumer, and the MS ASEs act as the supplier, of the MS abstract service.

These application-service-elements are in turn supported by other application-service-elements.

The remote operations service element (ROSE) supports the request/reply paradigm of the abstract operations that occur at the ports in the abstract model. The MSSE, MDSE, MRSE and MASE provide the mapping function of the abstract-syntax notation of an abstract-service onto the services provided by the ROSE.

Optionally, the reliable transfer service element (RTSE) may be used to reliably transfer the application-protocol-data-units (APDUs) that contain the parameters of the operations between AEs.

The association control service element (ACSE) supports the establishment and release of an application-association between a pair of AEs. Associations between an MTS-user and the MTS may be established by either the MTS-user or the MTS. Associations between an MS-user and an MS may be established only by the MS-user. Only the initiator of an established association can release it.

The combination of one or more of the MSSE, MDSE, MRSE and MASE, together with their supporting ASEs, defines the application-context of an application-association. Note that a single application-association may be used to support one or more types paired between two objects in the abstract model.

Table 1/X.419 identifies the application-contexts defined in this Recommendation for the MTS access protocol and MS access protocol.

If the MTS access protocol (P3) is supported, then support for the **mts-access** and **mts-forced-access** application-contexts is mandatory for an **MTA**. If an MTA supports the **mts-reliable-access** application-context, it shall also support the **mts-forced-reliable-access**, and vice versa. Support for each of the MTS access protocol (P3) application-context is optional for an MTS-user.

If the MS access protocol (P7) is supported, then support for the **ms-access** application-context is mandatory for an MS, and support for the **ms-reliable-access** application-context is optional. Support for each of the MS access protocol (P7) application-contexts is optional for an MS-user.

Figure 1/X.419 models an application-context between an MTS-user and the MTS. The consumer role of the MTS-user ASEs, and the supplier role of the MTS ASEs, is indicated by a subscripted "c" or "s", respectively.

Similarly, Figure 2/X.419 models an application-context between an MS-user and the MS.

TABLE 1/X.419

MHS access protocol application contexts

Application context	Message Handling ASEs				Supporting ASEs		
	MSSE	MDSE	MRSE	MASE	ROSE	RTSE	ACSE
MTS access protocol							
mts-access	C	C	–	C	X	–	X
mts-forced-access	S	S	–	S	X	–	X
mts-reliable-access	C	C	–	C	X	X	X
mts-forced-reliable-access	S	S	–	S	X	X	X
MS access protocol							
ms-access	C	–	C	C	X	–	X
ms-reliable-access	C	–	C	C	X	X	X

X present

– absent

C present with initiator the consumer

S present with initiator the supplier

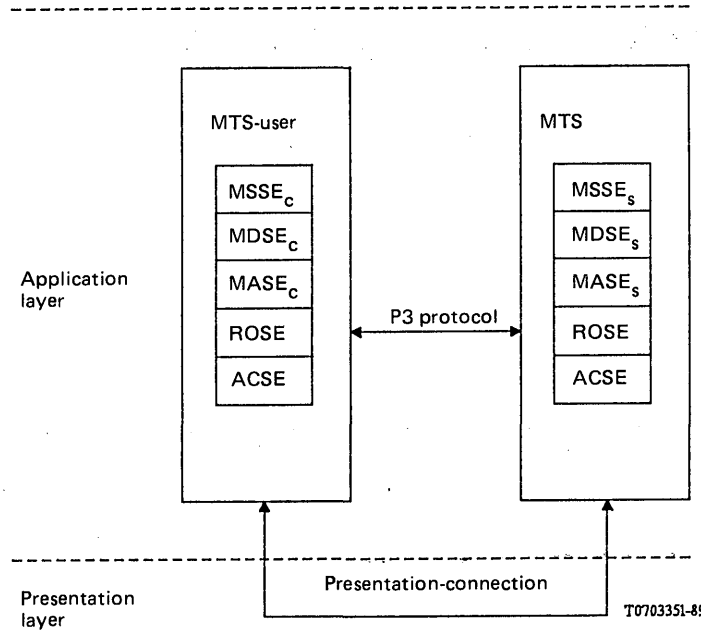


FIGURE 1/X.419

MTS access protocol model

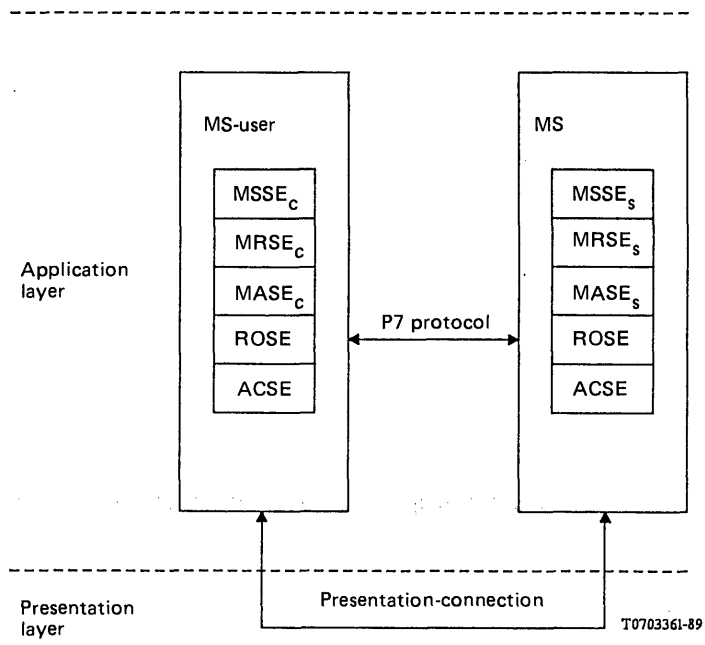


FIGURE 2/X.419

An MS access protocol model

6.2 *Services provided by the MTS access protocol*

The MTS access protocol (P3) comprises the following operations which provide the services defined in Recommendation X.411:

MTS-bind and MTS-unbind

a) MTS-bind

b) MTS-unbind

Message submission service element (MSSE)

c) message-submission

d) probe-submission

e) cancel-deferred-delivery

f) submission-control

Message delivery service element (MDSE)

g) message-delivery

h) report-delivery

i) delivery-control

Message administration service element (MASE)

j) register

k) change-credentials.

6.3 *Services provided by the MS access protocol*

The MS access protocol (P7) comprises the following operations which provide the services defined in Recommendation X.413:

MS-bind and MS-unbind

a) MS-bind

b) MS-unbind

Message submission service element (MSSE)

c) message-submission

d) probe-submission

e) cancel-deferred-delivery

f) submission-control

Message retrieval service element (MRSE)

g) summarize

h) list

i) fetch

j) delete

k) register-MS

l) alert

Message administration service element (MASE)

m) register

n) change-credentials.

6.4 *Use of underlying services*

The MHS access protocols make use of underlying services as described below.

6.4.1 *Use of ROSE services*

The remote operations service element (ROSE) is defined in Recommendation X.219.

The ROSE supports the request/reply paradigm of remote operations.

The MSSE, MDSE, MRSE and MASE are the sole users of the RO-INVOKE, RO-RESULT, RO-ERROR, RO-REJECT-U and RO-REJECT-P services of the ROSE.

The remote operations of the MTS access protocol (P3) and the MS access protocol (P7) are Class 2 (asynchronous) operations.

6.4.2 *Use of RTSE services*

The reliable transfer service element (RTSE) is defined in Recommendation X.218.

The RTSE provides for the reliable transfer of application-protocol-data units (APDUs). The RTSE ensures that each APDU is completely transferred exactly once, or that the sender is warned of an exception. The RTSE recovers from communication and end-system failure and minimizes the amount of retransmission needed for recovery.

Alternative application-contexts with and without RTSE are defined to support the MHS access protocols.

The RTSE is used in the normal mode. The use of the normal mode of the RTSE implies the use of the normal mode of the ACSE and the normal mode of the presentation-service.

If the RTSE is included in an application-context, the MHS access protocol MTS-bind and MTS-unbind (or MS-bind and MS-unbind) are the sole users of the RT-OPEN and RT-CLOSE services of the RTSE. The ROSE is the sole user of the RT-TRANSFER, RT-TURN-PLEASE, RT-TURN-GIVE, RT-P-ABORT and RT-U-ABORT services of the RTSE.

6.4.3 *Use of ACSE services*

The association control service element (ACSE) is defined in Recommendation X.217.

The ACSE provides for the control (establishment, release, abort) of application-associations between AEs.

If the RTSE is not included in an application-context, the MHS access protocol MTS-bind and MTS-unbind (or MS-bind and MS-unbind) are the sole users of the A-ASSOCIATE and A-RELEASE services of the ACSE in normal mode. The ROSE is the user of the A-ABORT and A-P-ABORT services of the ACSE.

If the RTSE is included in an application-context, the RTSE is the sole user of the A-ASSOCIATE, A-RELEASE, A-ABORT and A-P-ABORT services of the ACSE. The use of the normal mode of the RTSE implies the use of the normal mode of the ACSE and the normal mode of the presentation-service.

6.4.4 *Use of the presentation-service*

The presentation-service is defined in Recommendation X.216.

The presentation layer coordinates the representation (syntax) of the application layer semantics that are to be exchanged.

In normal mode, a different presentation-context is used for each abstract-syntax included in the application-context.

The ACSE is the sole user of the P-CONNECT, P-RELEASE, P-U-ABORT and P-P-ABORT services of the presentation-service.

If the RTSE is not included in the application-context, the ROSE is the sole user of the P-DATA service of the presentation-service.

If the RTSE is included in the application-context, the RTSE is the sole user of the P-ACTIVITY-START, P-DATA, P-MINOR-SYNCHRONIZE, P-ACTIVITY-END, P-ACTIVITY-INTERRUPT, P-ACTIVITY-DISCARD, P-U-EXCEPTION-REPORT, P-ACTIVITY-RESUME, P-P-EXCEPTION-REPORT, P-TOKEN-PLEASE and P-CONTROL-GIVE services of the presentation-service. The use of the normal mode of the RTSE implies the use of the normal mode of the ACSE and the normal mode of the presentation-service.

6.4.5 *Use of lower layer services*

The session-service is defined in Recommendation X.215. The session layer structures the dialogue of the flow of information between the end-systems.

If the RTSE is included in the application-association, the kernel, half-duplex, exceptions, minor-synchronize and activity-management functional units of the session-service are used by the presentation layer.

If the RTSE is not included in the application-association, the kernal and duplex functional units of the session-service are used by the presentation layer.

The transport-service is defined in Recommendation X.214. The transport layer provides for the end-to-end transparent transfer of data over the underlying network connection.

The choice of the class of transport-service used by the session layer depends on the requiements for multiplexing and error recovery. Support for transport class 0 (non-multiplexing) is mandatory. Transport expedited service is not used.

Support for other classes is optional. A multiplexing class may be used to multiplex an MHS access protocol and other access protocols (e.g. the directory access protocol (DAP) defined in Recommendation X.519) over the same network connection. An error recovery class may be chosen if the RTSE is omitted from an application-context over a network connection with an unacceptable residual error rate.

An underlying network supporting the OSI network-service defined in Recommendation X.213 is assumed.

A network-address is as defined in Recommendation X.121, Recommendations E.163/E.164, or Recommendation X.200 (OSI NSAP-address).

7 MTS access protocol abstract syntax definition

The abstract-syntax of the MTS access protocol (P3) is defined in Figure 3/X.419.

The abstract-syntax of the MTS access protocol (P3) is defined using the abstract-syntax notation (ASN.1) defined in Recommendation X.208, and the remote operations notation defined in Recommendation X.219.

The abstract-syntax definition of the MTS access protocol (P3) has the following major parts:

- *Prologue:* declarations of the exports from, and imports to, the MTS Access Protocol (P3) module (Figure 3/X.419, Part 1).
- *Application contexts:* definitions of application-contexts that may be used between an MTS-user and the MTS (Figure 3/X.419, Parts 2 and 3).
- *Message submission service element:* definitions of the message submission service element (MSSE) and its remote operations and errors (Figure 3/X.419, Part 4).
- *Message delivery service element:* definitions of the message delivery service element (MDSE) and its remote operations and errors (Figure 3/X.419, Part 5).
- *Message administration service element:* definitions of the message administration service element (MSSE) and its remote operations and errors (Figure 3/X.419, Part 6).

```
MTSAccessProtocol { joint-iso-ccitt mhs-motis(6) protocols(0) modules(0) mts-access-protocol(1) }

DEFINITIONS IMPLICIT TAGS ::=

BEGIN


-- Prologue

EXPORTS
        -- Application service elements
        mSSE, mDSE, mASE;

IMPORTS
        -- Application service elements and application contexts
        APPLICATION-SERVICE-ELEMENT, APPLICATION-CONTEXT, aCSE
                FROM Remote-Operations-Notation-extension { joint-iso-ccitt remote-operations(4)
                    notation-extension(2) }

        rTSE
                FROM Reliable-Transfer-APDUs { joint-iso-ccitt reliable-transfer(3) apdus(0) }

        -- MTS abstract service parameters
        MTSBind, MTSUnbind, MessageSubmission, ProbeSubmission, CancelDeferredDelivery,
        SubmissionControl, MessageDelivery, ReportDelivery, DeliveryControl, Register,
        ChangeCredentials, SubmissionControlViolated, ElementOfServiceNotSubscribed,
        DeferredDeliveryCancellationRejected, OriginatorInvalid, RecipientImproperlySpecified,
        MessageSubmissionIdentifierInvalid, InconsistentRequest, SecurityError,
        UnsupportedCriticalFunction, RemoteBindError, DeliveryControlViolated, ControlViolatesRegistration,
        RegisterRejected, NewCredentialsUnacceptable, OldCredentialsIncorrectlySpecified
                FROM MTSAbstractService { joint-iso-ccitt mhs-motis(6) mts(3) modules(0)
                    mts-abstract-service(1)}

        -- Object identifiers
        id-ac-mts-access, id-ac-mts-forced-access, id-ac-mts-reliable-access, id-ac-mts-forced-reliable-access,
        id-as-acse, id-as-msse, id-as-mdse, id-as-mrse, id-as-mase, id-as-mts, id-as-mts-rtse,
        id-ase-msse, id-ase-mdse, id-ase-mase
                FROM MHSProtocolObjectIdentifiers { joint-iso-ccitt mhs-motis(6) protocols(0)
                    modules(0) object-identifiers(0) };
```

FIGURE 3/X.419 (Part 1 of 6)

Abstract syntax definition of the MTS access protocol (P3)

```
-- Application contexts omitting RTSE

-- MTS-user initiated

mts-access APPLICATION-CONTEXT
        APPLICATION SERVICE ELEMENTS { aCSE }
        BIND MTSBind
        UNBIND MTSUnbind
        REMOTE OPERATIONS { rOSE }
        INITIATOR CONSUMER OF { mSSE, mDSE, mASE }
        ABSTRACT SYNTAXES {
                id-as-acse,         -- of ACSE
                id-as-msse,         -- of MSSE, including ROSE
                id-as-mdse,         -- of MDSE, including ROSE
                id-as-mase,         -- of MASE, including ROSE
                id-as-mts           -- of MTSBind and MTSUnbind -- }
        ::= id-ac-mts-access


-- MTS initiated

mts-forced-access APPLICATION-CONTEXT
        APPLICATION SERVICE ELEMENTS { aCSE }
        BIND MTSBind
        UNBIND MTSUnbind
        REMOTE OPERATIONS { rOSE }
        RESPONDER CONSUMER OF { mSSE, mDSE, mASE }
        ABSTRACT SYNTAXES {
                id-as-acse,         -- of ACSE
                id-as-msse,         -- of MSSE, including ROSE
                id-as-mdse,         -- of MDSE, including ROSE
                id-as-mase,         -- of MASE, including ROSE
                id-as-mts           -- of MTSBind and MTSUnbind -- }
        ::= id-ac-mts-forced-access
```

FIGURE 3/X.419 (Part 2 to 6)

Abstract syntax definition of the MTS access protocol (P3)

```
-- Application contexts including RTSE in normal mode

-- MTS-user initiated

mts-reliable-access APPLICATION-CONTEXT
        APPLICATION SERVICE ELEMENTS { aCSE, rTSE }
        BIND MTSBind
        UNBIND MTSUnbind
        REMOTE OPERATIONS { rOSE }
        INITIATOR CONSUMER OF { mSSE, mDSE, mASE }
        ABSTRACT SYNTAXES {
                id-as-acse,         -- of ACSE
                id-as-msse,         -- of MSSE, including ROSE
                id-as-mdse,         -- of MDSE, including ROSE
                id-as-mase,         -- of MASE, including ROSE
                id-as-mts-rtse      -- of MTSBind and MTSUnbind, including RTSE -- }
        ::= id-ac-mts-reliable-access


-- MTS initiated

mts-forced-reliable-access APPLICATION-CONTEXT
        APPLICATION SERVICE ELEMENTS { aCSE, rTSE }
        BIND MTSBind
        UNBIND MTSUnbind
        REMOTE OPERATIONS { rOSE }
        RESPONDER CONSUMER OF { mSSE, mDSE, mASE }
        ABSTRACT SYNTAXES {
                id-as-acse,         -- of ACSE
                id-as-msse,         -- of MSSE, including ROSE
                id-as-mdse,         -- of MDSE, including ROSE
                id-as-mase,         -- of MASE, including ROSE
                id-as-mts-rtse      -- of MTSBind and MTSUnbind, including RTSE -- }
        ::= id-ac-mts-forced-reliable-access
```

FIGURE 3/X.419 (Part 3 of 6)

Abstract syntax definition of the MTS access protocol (P3)

```
-- Message submission service element

mSSE APPLICATION-SERVICE-ELEMENT
        CONSUMER INVOKES {
                message-submission,
                probe-submission,
                cancel-deferred-delivery }
        SUPPLIER INVOKES {
                submission-control }
        ::= id-ase-msse


-- Remote operations

message-submission MessageSubmission ::= 3

probe-submission ProbeSubmission ::= 4

cancel-deferred-delivery CancelDeferredDelivery ::= 7

submission-control SubmissionControl ::= 2


-- Remote errors

submission-control-violated SubmissionControlViolated ::= 1

element-of-service-not-subscribed ElementOfServiceNotSubscribed ::= 4

deferred-delivery-cancellation-rejected DeferredDeliveryCancellationRejected ::= 8

originator-invalid OriginatorInvalid ::= 2

recipient-improperly-specified RecipientImproperlySpecified ::= 3

message-submission-identifier-invalid MessageSubmissionIdentifierInvalid ::= 7

inconsistent-request InconsistentRequest ::= 11

security-error SecurityError ::= 12

unsupported-critical-function UnsupportedCriticalFunction ::= 13

remote-bind-error RemoteBindError ::= 15
```

FIGURE 3/X.419 (Part 4 of 6)

Abstract syntax definition of the MTS access protocol (P3)

```
-- Message delivery service element

mDSE APPLICATION-SERVICE-ELEMENT
        CONSUMER INVOKES {
                delivery-control}
        SUPPLIER INVOKES {
                message-delivery,
                report-delivery }
        ::= id-ase-mdse


-- Remote operations

message-delivery MessageDelivery ::= 5

report-delivery ReportDelivery ::= 6

delivery-control DeliveryControl ::= 2


-- Remote errors

delivery-control-violated DeliveryControlViolated ::= 1

control-violates-registration ControlViolatesRegistration ::= 14


-- security-error ::= 12, defined in Part 4

-- unsupported-critical-function ::= 13, defined in Part 4
```

FIGURE 3/X.419 (Part 5 of 6)

Abstract syntax definition of the MTS access protocol (P3)

```
-- Message administration service element

mASE APPLICATION-SERVICE-ELEMENT
        CONSUMER INVOKES {
                register,
                change-credentials }
        SUPPLIER INVOKES {
                change-credentials }
        ::= id-ase-mase


-- Remote operations

register Register ::= 1

change-credentials ChangeCredentials ::= 8


-- Remote errors

register-rejected RegisterRejected ::= 10

new-credentials-unacceptable NewCredentialsUnacceptable ::= 6

old-credentials-incorrectly-specified OldCredentialsIncorrectlySpecified ::= 5


END -- of MTSAccessProtocol
```

FIGURE 3/X.419 (Part 6 of 6)

Abstract syntax definition of the MTS access protocol (P3)

8 MS access protocol abstract syntax definition

The abstract-syntax of the MS access protocol (P7) is defined in Figure 4/X.419.

The abstract-syntax of the MS access protocol (P7) is defined using the abstract syntax notation (ASN.1) defined in Recommendation X.208, and the remote operations notation defined in Recommendation X.219.

The abstract-syntax definition of the MS access protocol (P7) has the following major parts:

- *Prologue:* declarations of the exports from, and imports to, the MTS access protocol (P3) module (Figure 4/X.419, Part 1).
- *Application contexts:* definitions of application-contexts that may be used between an MS-user and the MS (Figure 4/X.419, Part 2).
- *Message retrieval service element:* definitions of the message retrieval service element (MRSE) and its remote operations and errors (Figure 4/X.419, Parts 3 and 4).

```
MSAccessProtocol { joint-iso-ccitt mhs-motis(6) protocols(0) modules(0) ms-access-protocol(2) }

DEFINITIONS IMPLICIT TAGS ::=

BEGIN


-- Prologue

EXPORTS
        mRSE;

IMPORTS
        -- Application service elements and application contexts
        APPLICATION-SERVICE-ELEMENT, APPLICATION-CONTEXT, aCSE
                FROM Remote-Operations-Notation-extension { joint-iso-ccitt remote-operations(4)
                    notation-extension(2) }

        rTSE
                FROM Reliable-Transfer-APDUs { joint-iso-ccitt reliable-transfer(3) apdus(0) }

        mSSE, mASE
                FROM MTSAccessProtocol { joint-iso-ccitt mhs-motis(6) protocols(0)
                modules(0) mts-access-protocol(1)}

        -- MS abstract service parameters
        MSBind, MSUnbind, Summarize, List, Fetch, Delete, Register-MS, Alert, AttributeError,
        AutoActionRequestError, DeleteError, FetchRestrictionError, RangeError, SecurityError,
        ServiceError, SequenceNumberError
                FROM MSAbstractService { joint-iso-ccitt mhs-motis(6) ms(4) modules(0)
                    abstract-service(1)}

        -- Object identifiers
        id-ac-ms-access, id-ac-ms-reliable-access, id-as-acse, id-as-msse, id-as-mrse, id-as-mase, id-as-ms, id-as-ms-rtse,
        id-ase-mrse
                FROM MHSProtocolObjectIdentifiers { joint-iso-ccitt mhs-motis(6) protocols(0)
                    modules(0) object-identifiers(0) };
```

FIGURE 4/X.419 (Part 1 of 4)

Abstract syntax definition of the MS access protocol (P7)

```
-- Application context omitting RTSE

ms-access APPLICATION-CONTEXT
        APPLICATION SERVICE ELEMENTS { aCSE }
        BIND MSBind
        UNBIND MSUnbind
        REMOTE OPERATIONS { rOSE }
        INITIATOR CONSUMER OF { mSSE, mRSE, mASE }
        ABSTRACT SYNTAXES {
                id-as-acse,         -- of ACSE
                id-as-msse,         -- of MSSE, including ROSE
                id-as-mrse,         -- of MRSE, including ROSE
                id-as-mase,         -- of MASE, including ROSE
                id-as-ms            -- of MSBind and MSUnbind -- }
        ::= id-ac-ms-access


-- App;ication context including RTSE

ms-reliable-access APPLICATION-CONTEXT
        APPLICATION SERVICE ELEMENTS { aCSE, rTSE }
        BIND MSBind
        UNBIND MSUnbind
        REMOTE OPERATIONS { rOSE }
        INITIATOR CONSUMER OF { mSSE, mRSE, mASE }
        ABSTRACT SYNTAXES {
                id-as-acse,         -- of ACSE
                id-as-msse,         -- of MSSE, including ROSE
                id-as-mrse,         -- of MRSE, including ROSE
                id-as-mase,         -- of MASE, including ROSE
                id-as-ms-rtse       -- of MSBind and MSUnbind, including RTSE -- }
        ::= id-ac-ms-reliable-access
```

FIGURE 4/X.419 (Part 2 of 4)

Abstract syntax definition of the MS access protocol (P7)

```
-- Message retrieval service element

mRSE APPLICATION-SERVICE-ELEMENT
        CONSUMER INVOKES {
                summarize,
                list,
                fetch,
                delete,
                register-MS, }
        SUPPLIER INVOKES {
                alert }
        ::= id-ase-mrse


-- Remote operations

summarize Summarize ::= 20

list List ::= 21

fetch Fetch ::= 22

delete Delete ::= 23

register-ms Register-MS::= 24

alert Alert ::= 25


-- Remote errors

attribute-error AttributeError ::= 21

auto-action-request-error AutoActionRequestError ::= 22

delete-error DeleteError ::= 23

fetch-restriction-error FetchRestrictionError ::= 24

range-error RangeError ::= 25

security-error SecurityError ::= 26

service-error ServiceError ::= 27
```

FIGURE 4/X.419 (Part 3 of 4)

Abstract syntax definition of the MS access protocol (P7)

```
sequence-number-error SequenceNumberError ::= 28


END  -- of MSAccessProtocol
```

FIGURE 4/X.419 (Part 4 of 4)

Abstract syntax definition of the MS access protocol (P7)

9 Mapping onto used services

This paragraph defines the mapping of the MHS access protocols onto the used services.

Paragraph 9.1 defines the mapping onto used services for application-contexts that omit the RTSE. Paragraph 9.2 defines the mapping onto used services for application contexts that include the RTSE.

9.1 *Application-contexts omitting RTSE*

This paragraph defines the mapping of the MHS access protocols onto the used services for application-contexts that omit the RTSE. Support for this mapping is optional for conformance to this Recommendation.

9.1.1 *Mapping onto ACSE*

This paragraph defines the mapping of the abstract-bind (MTS-bind or MS-bind) and abstract-unbind (MTS-unbind or MS-unbind) services onto the services of the ACSE in normal mode for application-contexts that omit the RTSE. The ACSE is defined in Recommendation X.217.

9.1.1.1 *Abstract-bind onto A-ASSOCIATE*

The abstract-bind service is mapped onto the A-ASSOCIATE service of the ACSE. The use of the parameters of the A-ASSOCIATE service is qualified in the following paragraphs.

9.1.1.1.1 *Mode*

This parameter shall be supplied by the initiator of the association in the A-ASSOCIATE request primitive, and shall have the value "normal mode".

9.1.1.1.2 *Application context name*

The initiator of the association shall propose one of the application-contexts defined in this Recommendation that omit the RTSE in the A-ASSOCIATE request primitive (see Table 1/X.419).

9.1.1.1.3 *User information*

The mapping of the bind-operation of the abstract-bind service onto the user information parameter of the A-ASSOCIATE request primitive is defined in Recommendation X.219.

9.1.1.1.4 *Presentation context definition list*

The initiator of the association shall supply the presentation context definition list in the A-ASSOCIATE request primitive.

The presentation context definition list comprises a presentation-context-definition for each abstract-syntax included in the application-context. A presentation-context-definition comprises a presentation-context-identifier and an abstract-syntax-name for the ASE. Each named abstract syntax for the MSSE, MDSE, MRSE and MASE includes the ROSE APDUs.

Paragraphs 7 and 8 define the abstract-syntaxes included in the application-contexts.

9.1.1.1.5 *Quality of service*

This parameter shall be supplied by the initiator of the association in the A-ASSOCIATE request primitive, and by the responder of the association in the A-ASSOCIATE response primitive. The parameters "extended control" and "optimized dialogue transfer" shall be set to not required. The remaining parameters shall be such that default values are used.

9.1.1.1.6 *Session requirements*

This parameter shall be set by the initiator of the association in the A-ASSOCIATE request primitive, and by the responder of the association in the A-ASSOCIATE response primitive. The parameter shall be set to specify the following functional units:

a) kernel

b) duplex.

9.1.1.2 *Abstract-unbind onto A-RELEASE*

The abstract-unbind service is mapped onto the A-RELEASE service of the ACSE. The use of the parameters of the A-RELEASE service is qualified in the following paragraphs.

9.1.1.2.1 *Result*

This parameter shall have the value "affirmative".

9.1.1.3 *Use of A-ABORT and A-P-ABORT services*

The ROSE is the user of the A-ABORT and A-P-ABORT servicesof the ACSE.

9.1.2 *Mapping onto ROSE*

The MSSE, MDSE, MRSE and MASE services are mapped onto the RO-INVOKE, RO-RESULT, RO-ERROR, RO-REJECT-U and RO-REJECT-P services of the ROSE. The mapping of the abstract-syntax notation of the MSSE, MDSE, MRSE and MASE onto the ROSE services is as defined in Recommendation X.219.

9.2 *Application-contexts including RTSE*

This paragraph defines the mapping of the MHS access protocols onto the used services for application-contexts that include the RTSE in normal mode. Support for this mapping is optional for conformance to this Recommendation. No mappings are defined onto the RTSE in X.410-1984 mode. The RTSE is defined in Recommendation X.218.

9.2.1 *Mappint onto RT-OPEN and RT-CLOSE*

This paragraph defines the mapping of the abstract-bind (MTS-bind or MS-bind) and abstract-unbind (MTS-unbind or MS-unbind) services onto the RT-OPEN and RT-CLOSE services of the RTSE in normal mode.

9.2.1.1 *Abstract-bind onto RT-OPEN*

The abstract-bind service is mapped onto the RT-OPEN service of the RTSE. The use of the parameters of the RT-OPEN service is qualified in the following paragraphs.

9.2.1.1.1 *Mode*

This parameter shall be supplied by the initiator of the association in the RT-OPEN request primitive, and shall have the value "normal mode".

9.2.1.1.2 *Application context name*

The initiator of the association shall propose one of the application-contexts defined in this Recommendation that include the RTSE in normal mode in the RT-OPEN request primitive (see Table 1/X.419).

9.2.1.1.3 *User-data*

The mapping of the bind-operation of the abstract-bind service onto the user-data parameter of the RT-OPEN request primitive is defined in Recommendation X.219.

9.2.1.1.4 *Presentation context definition list*

The initiator of the association shall supply the presentation context definition list in the RT-OPEN request primitive.

The presentation context definition list comprises a presentation-context-definition for each abstract-syntax included in the application context. A presentation-context-definition comprises a presentation-context-identifier and an abstract-syntax-name for the ASE. Each named abstract-syntax for the MSSE, MDSE, MRSE and MASE includes the ROSE APDUs. The named abstract-syntax for the RTSE includes the abstract-syntax for the bind-operation of the abstract-bind service.

Paragraphs 7 and 8 define the abstract-syntaxes included in the application-contexts.

9.2.1.2 *Abstract-unbind onto RT-CLOSE*

The abstract-unbind service is mapped onto the RT-CLOSE service of the RTSE.

9.2.2 *Mapping onto ROSE*

The MSSE, MDSE and MASE services are mapped onto the RO-INVOKE, RO-RESULT, RO-ERROR, RO-REJECT-U and RO-REJECT-P services of the ROSE. The mapping of the abstract-syntax notation of the MSSE, MDSE, MRSE and MASE onto the ROSE services is performed as defined in Recommendation X.219.

ROSE is the user of the RT-TRANSFER, RT-TURN-PLEASE, RT-TURN-GIVE, RT-P-ABORT and RT-U-ABORT services of the RTSE. The use of the RTSE services by the ROSE is defined in Recommendation X.229.

9.2.2.1 *Managing the turn*

Recommendation X.229 defines the use by the ROSE of the RT-TURN-PLEASE and RT-TURN-GIVE services of the RTSE to manage the turn.

Table 2/X.419 defines the values of the priority parameter of the RT-TURN-PLEASE service used by the ROSE to request the turn.

Priority zero is the highest priority, and is reserved for the action of releasing the association by the initiator.

Priority one is used by the ROSE for the RORJ APDU and ROER APDU to provide the RO-REJECT-U and RO-ERROR services of the ROSE.

Priority two is used by the ROSE for the RORS APDU to provide the RO-RESULT services of the ROSE.

Priority three to seven shall be used for the ROIV APDU to provide the RO-INVOKE service for the MHS access protocol remote operations. In the case of a remote operation whose arguments include a message, the ROIV APDU is prioritized as a function of the **priority** of the message – **urgent, normal** or **non-urgent**.

TABLE 2/X.419

Remote operation priorities

Priority	MSSE	MDSE	MRSE	MASE
0	Association release			
1	RO-REJECT-U RO-ERROR			
2	RO-RESULT			
3	Submission-control	Delivery-control		
4	Message-submission (urgent)	Message-delivery (urgent)	Alert	
5	Probe-submission	Report-delivery	Register-MS Summarize List Fetch Delete	Register Change-credentials
6	Message-submission (normal)	Message-delivery (normal)		
7	Message-submission (non-urgent)	Message-delivery		

10 Conformance

A system (UA, MS or MTA) claiming conformance to the MHS access protocols specified in this Recommendation shall comply with the requirements in §§ 10.1, 10.2 and 10.3.

10.1 *Statement requirements*

The following shall be stated:

a) the type of system for which conformance is claimed (UA, MS, MTA or MTA/MS);

b) the application-contexts defined in Section 2 of this Recommendation for which conformance is claimed.

Conformance can be claimed to the MTS access protocol (P3), or the MS access protocol (P7), or both. Table 3/X.419 classifies the support for application-contexts required for conformance to the MTS access protocol (P3). Table 4/X.419 classifies the support for application-contexts required for conformance to the MS access protocol (P7).

TABLE 3/X.419

MTS access protocol conformance requirements

Application context	MTA	MTS-user
MTS access protocol		
mts-access	Mandatory	Optional
mts-forced-access	Mandatory	Optional
mts-reliable-access	Optional (see note)	Optional
mts-forced-reliable-access	Optional (see note)	Optional

Note – If an MTA claims conformance to the mts-reliable-access application-context, it shall also claim conformance to the mts-forced-reliable-access application-context, and vice versa.

TABLE 4/X.419

MS access protocol conformance requirements

Application context	MS	MS-user
MS access protocol		
ms-access	Mandatory	Optional
ms-reliable-access	Optional	Optional

10.2 *Static requirements*

The system shall:

a) conform to the abstract-syntax definition(s) of the MHS access protocols defined in §§ 7 and 8 of this Recommendation, required by the application-contexts for which conformance is claimed.

10.3 *Dynamic requirements*

The system shall:

a) conform to the mapping onto used services defined in § 9 of this Recommendation, required by the application-contexts for which conformance is claimed;

b) conform to the use of underlying services defined in § 6.4 of this Recommendation.

SECTION 3 – MESSAGE TRANSFER SYSTEM TRANSFER PROTOCOL SPECIFICATION

11 Overview of the MTS transfer protocol

11.1 *Model*

Paragraph 10 of Recommendation X.411 refines the abstract model of the message transfer system (MTS), first presented in § 6 of that Recommendation, to reveal that the MTS object comprises a collection of message-transfer-agent (MTA) objects, which cooperate together to form the MTS and offer the MTS abstract service to its users.

In the refined abstract model, interactions between MTAs are modelled as a set of abstract operations which occur at the transfer-port paired between MTAs.

This paragraph describes how the MTA abstract service is supported by instances of OSI communication when the MTAs are realised as application-processes located in different open systems.

In the OSI environment, communication between application-processes is represented in terms of communication between a pair of application-entities (AEs) using the presentation-service. The functionality of an AE is factored into a set of one or more application-service-elements (ASEs). The interaction between AEs is described in terms of their use of the services provided by the ASEs.

The transfer-port services of the abstract model are supported by an application-service-element – the message transfer service element (MTSE), which in turn is supported by two other application-service-elements – the reliable transfer service element (RTSE) and the association control service element (ACSE).

The reliable transfer service element (RTSE) is used to reliably transfer application-protocol-data-units (APDUs) that contain the message, probes and reports between AEs.

The association control service element (ACSE) supports the establishment and release of an application-association between a pair of AEs. Associations between MTAs can be established by either MTA. Only the initiator of an established association can release it.

The combination of the MTSE, the RTSE and the ACSE defines the application-context of an application-association.

Figure 4/X.419 models the application-context between MTAs.

Three application-contexts are defined for the MTS transfer protocol as identified in Table 5/X.419.

TABLE 5/X.419

MTS transfer protocol application contexts

Application context	P1	RTSE mode
mts-transfer-protocol-1984	P1 1984	X.410-1984
mts-transfer-protocol	P1 1988	X.410-1984
mts-transfer	P1 1988	normal

The **mts-transfer-protocol-1984** is defined for interworking with implementations of the 1984 Recommendation X.411. In this application-context, the abstract-syntax of the MTSE is constrained to that defined in the 1984 Recommendation X.411. These constraints are identified by underlining of the 1988 extensions to the abstract syntax of the MTSE in the defining ASN.1 module in Recommendation X.411. The changes are also listed in Annex C of this Recommendation for reference. The **mts-transfer-protocol-1984** is supported by the RTSE in X.410-1984 mode. Support for the **mts-transfer-protocol-1984** is mandatory for conformance to this Recommendation.

The **mts-transfer-protocol** is defined to enable interworking between implementations which support the 1988 extended functionality via systems which have had a minimal upgrade from conformance to the 1984 Recommendation X.411. The **mts-transfer-protocol** provides for controlled transparency of the upgraded system to the 1988 extensions. The **mts-transfer-protocol** is supported by the RTSE in X.410-1984 mode. Support for the **mts-transfer-protocol** is mandatory for conformance to this Recommendation.

The **mts-transfer** application-context is supported by the RTSE in normal mode. It is envisaged that, over time, most systems will migrate to support the **mts-transfer** application-context. Support for the **mts-transfer** application-context is optional for conformance to this Recommendation. Note that in ISO 10021-6 support for the **mts-transfer** application-context is mandatory. A future version of this Recommendation is likely to make support for the **mts-transfer** application-context mandatory as part of a migration strategy to enable support for extended functionaility and to maximise interworking.

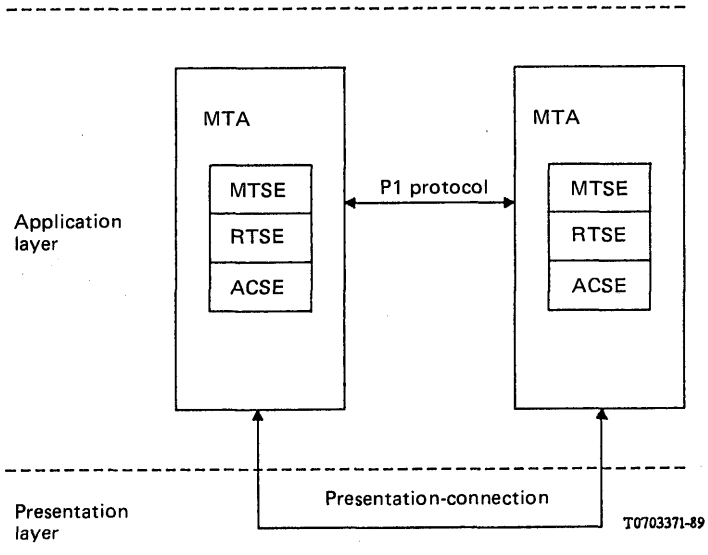


FIGURE 4/X.419

MTS transfer protocol model

11.2 *Services provided by the MTS transfer protocol*

The MTS transfer protocol (P1) provides the following services defined in Recommendation X.411:

MTA-bind and MTA-unbind

a) MTA-bind

b) MTA-unbind

Message transfer service element (MTSE)

c) message-transfer

d) probe-transfer

e) report-transfer

11.3 *Use of underlying services*

The MTS transfer protocol (P1) makes use of underlying services as described below.

11.3.1 *Use of the RTSE services*

The reliable transfer service element (RTSE) is defined in Recommendation X.218.

The RTSE provides for the reliable transfer of application-protocol-data-units (APDUs). The RTSE ensures that each APDU is completely transferred once, or that the sender is warned of an exception. The RTSE recovers from communication and end-system failure and minimises the amount of retransmission needed for recovery.

The RTSE services are used to support the MTS transfer protocol (P1). Support for RTSE in X.410-1984 mode is mandatory. Support for the RTSE in normal mode is optional. Note that in ISO 10021-6, support for the RTSE in normal mode is mandatory, and support for the RTSE in X.410-1984 is optional.

The use of the X.410-1984 mode of the RTSE implies the use of the X.410-1984 mode of the ACSE and the X.410-1984 mode of the presentation-service. The use of the normal mode of the RTSE implies the use of the normal mode of the ACSE and the normal mode of the presentation-service.

The MTS transfer protocol (P1) is the sole user of the RT-OPEN, RT-CLOSE, RT-TRANSFER, RT-TURN-PLEASE, RT-TURN-GIVE, RT-P-ABORT and RT-U-ABORT services of the RTSE.

11.3.2 *Use of the ACSE services*

The association control service element (ACSE) is defined in Recommendation X.217.

The ACSE provides for the control (establishment, release, abort) of application-associations between AEs.

The RTSE is the sole user of the A-ASSOCIATE, A-RELEASE, A-ABORT and A-P-ABORT services of the ACSE. The use of the X.410-1984 mode of the RTSE implies the use of the X.410-1984 mode of the ACSE and the X.410-1984 mode of the presentation-service. The use of the normal mode of the RTSE implies the use of the normal mode of the ACSE and the normal mode of the presentation-service.

11.3.3 *Use of the presentation-service*

The presentation-service is defined in Recommendation X.216.

The presentation layer coordinates the representation (syntax) of the application layer semantics that are to be exchanged.

In X.410-1984 mode, a single default presentation-context is used for the underlying presentation-connection. This presentation-context includes a single abstract-syntax for all of the ASEs included in the application-context (i.e. MTSE, RTSE and ACSE).

In normal mode, a different presentation-context is used for each abstract-syntax included in the application-context.

Presentation layer addressing is not used for the message transfer protocol (P1) in X.410-1984 mode.

The ACSE is the sole user of the P-CONNECT, P-RELEASE, P-U-ABORT and P-P-ABORT services of the presentation-service.

The RTSE is the sole user of the P-ACTIVITY-START, P-DATA, P-MINOR-SYNCHRONIZE, P-ACTIVITY-END, P-ACTIVITY-INTERRUPT, P-ACTIVITY-DISCARD, P-U-EXCEPTION-REPORT, P-ACTIVITY-RESUME, P-P-EXCEPTION-REPORT, P-TOKEN-PLEASE, and P-CONTROL-GIVE services of the presentation service. The use of the X.410-1984 mode of the RTSE implies the use of the X.410-1984 mode of the ACSE and the X.410-1984 mode of the presentation-service. The use of the normal mode of the RTSE implies the use of the normal mode of the ACSE and the normal mode of the presentation-service.

11.3.4 *Use of lower layer services*

The session-service is defined in Recommendation X.215. The session layer structures the dialogue of the flow of information between the end-systems.

The use of the RTSE requires the use of the kernel, half-duplex, exceptions, minor-synchronize and activity-management functional units by the presentation layer.

Session layer addressing is not used for the MTS transfer protocol (P1) when the RTSE is used in X.410-1984 mode. That is, a session-address shall not be passed in the Connect SPDU of the session layer.

The transport-service is defined in Recommendation X.214. The transport layer provides for the end-to-end transparent transfer of data over the underlying network connection.

The choice of the class of transport-service used by the session layer depends on the requirements for multiplexing and error recovery. Support for Class 0 is mandatory. Transport expedited services is not used.

Support for other classes is optional. The use of an error recovery class together with the RTSE duplicates mechanisms for error recovery.

The transport-address comprises a network-address and a transport-service-access-point identifier (TSAP-identifier). The TSAO-identifier is carried in the transport layer protocol. When the RTSE is used in X.410-1984 mode, it consists of up to sixteen IA5 digits.

An underlying network supporting the OSI network-service defined in Recommendation X.213 is assumed.

A network-address is as defined in Recommendation X.121, Recommendations E.163/E.164, or Recommendation X.200 (OSI NSAP-address).

11.4 *Establishing and releasing associations*

Associations between two MTAs are created in accordance with bilateral agreements covering the following:

a) the maximum number of associations that may exist simultaneously;

b) whether monologue or two-way-alternate associations are used;

c) which application-context is used;

d) which MTA has responsibility for establishing the associations;

e) whether associations are permanently established or established and released as required.

12 MTS transfer protocol abstract syntax definition

The abstract-syntax of the MTS transfer protocol (P1) is defined in Figure 5/X.419.

The abstract-syntax of the MTS transfer protocol (P1) is defined using the abstract-syntax notation (ASN.1) defined in Recommendation X.208, and the remote operations notation defined in Recommendation X.219.

The abstract-syntax definition of the MTS transfer protocol (P1) has the following major parts:

- *Prologue:* declarations of the exports from, and imports to, the MTS transfer protocol (P1) module (Figure 5/X.419, Part 1).
- *Application contexts:* definitions of the application-contexts used between MTAs (Figure 5/X.419, Part 2).
- *Message transfer service element:* definitions of the message transfer service element (MTSE) (Figure 5/X.419, Part 3).
- *MTS application protocol data units:* definition of the MTS application-protocol-data-units (APDUs): message, probe and report (Figure 5/X.419, Part 3).

```
MTSTransferProtocol { joint-iso-ccitt mhs-motis(6) protocols(0) modules(0) transfer-protocol(3) }

DEFINITIONS IMPLICIT TAGS ::=

BEGIN


-- Prologue

EXPORTS;

IMPORTS
        -- Application service elements and application contexts
        APPLICATION-SERVICE-ELEMENT, APPLICATION-CONTEXT, aCSE
                FROM Remote-Operations-Notation-extension { joint-iso-ccitt remote-operations(4)
                    notation-extension(2) }
        rTSE
                FROM Reliable-Transfer-APDUs { joint-iso-ccitt reliable-transfer(3) apdus(0) }

        -- MTA transfer port abstract service parameters
        MTABind, MTAUnbind, Message, Probe, Report,
                FROM MTAAbstractService { joint-iso-ccitt mhs-motis(6) mts(3) modules (0)
                    mta-abstract-service(2) }

        -- Object identifiers
        id-ac-mts-transfer, id-as-acse, id-as-mta-rtse, id-as-mtse, id-ase-mtse
                FROM MHSProtocolObjectIdentifiers { joint-iso-ccitt mhs-motis(6) protocols(0)
                    modules(0) object-identifiers(0) }
```

FIGURE 5/X.419 (Part 1 of 3)

Abstract syntax definition of the MTS transfer protocol (P1)

```
-- Application context including RTSE in normal mode

mts-transfer APPLICATION-CONTEXT
        APPLICATION SERVICE ELEMENTS { aCSE, rTSE, mTSE }
        BIND MTABind
        UNBIND MTAUnbind
        ABSTRACT SYNTAXES {
                id-as-acse,         -- of ACSE
                id-as-mts-rtse,     -- of MTABind and MTAUnbind, including RTSE
                id-as-mtse          -- of MTSE-- }
        ::= id-ac-mts-transfer


-- Application context including RTSE in X.410-1984 mode

mts-transfer-protocol INTEGER ::= 12


-- Application context for interworking with 1984 P1

mts-transfer-protocol-1984 INTEGER ::= 1
```

FIGURE 5/X.419 (Part 2 of 3)

Abstract syntax definition of the MTS transfer protocol (P1)

```
-- Message transfer service element

mTSE APPLICATION-SERVICE-ELEMENT
        ::= id-ase-mtse


-- MTS application protocol data units

MTS-APDU ::= CHOICE {
        message [0] Message,
        probe [2] Probe,
        report [1] Report }


END  -- of MTSTransferProtocol
```

FIGURE 5/X.419 (Part 3 of 3)

Abstract syntax definition of the MTS transfer protocol (P1)

13 Mapping onto used services

This paragraph defines the mapping of the MTS transfer protocol (P1) onto the used services.

Paragraph 13.1 defines the mapping of the MTS transfer protocol (P1) onto used services for application-contexts that include the RTSE in X.410-1984 mode. Paragraph 13.2 defines the mapping of the MTS transfer protocol (P1) onto used services for application-contexts that include the RTSE in normal mode.

13.1 *Mapping onto RTSE X.410-1984 mode*

This paragraph defines the mapping of the MTS transfer protocol (P1) onto used services for application-contexts that include the RTSE in X.410-1984 mode. Support for this mapping is mandatory for conformance to this Recommendation.

Paragraph 13.1.1 defines the mapping of the MTA-bind and MTA-unbind services onto the RT-OPEN and RT-CLOSE services of the RTSE in X.410-1984 mode. Paragraph 13.1.2 defines the mapping of the message-transfer, probe-transfer and report-transfer services onto the RT-TRANSFER service of the RTSE. Paragraph 13.1.3 describes managing the turn using the RT-TURN-PLEASE and RT-TURN-GIVE services of the RTSE. Paragraph 13.1.4 defines the use of the RT-P-ABORT service of the RTSE. Paragraph 13.1.5 defines the use of the RT-U-ABORT service of the RTSE (not used in X.410-1984 mode).

13.1.1 *Mapping onto RT-OPEN and RT-CLOSE*

This paragraph defines the mapping of the MTA-bind and MTA-unbind services onto the RT-OPEN and RT-CLOSE services of the RTSE in X.410-1984 mode.

13.1.1.1 *MTA-bind onto RT-OPEN*

The MTA-bind service is mapped onto the RT-OPEN service of the RTSE. The use of the parameters of the RT-OPEN service is qualified in the following clauses.

13.1.1.1.1 *Application-protocol*

This parameter shall be supplied by the initiator of the association of the RT-OPEN request primitive, and shall have the value **mts-transfer-protocol** (an integer value of "12") or **mts-transfer-protocol-1984** (an integer value of "1").

13.1.1.1.2 *User-data*

The value of the type defined in the ARGUMENT clause of the MTA-bind service is mapped onto the user-data parameter of the RT-OPEN request primitive by the initiator of the association.

If the responder of the association supplies the result parameter of the RT-OPEN response primitive with the value "accepted", the value of the type defined in the RESULT clause of the MTA-bind service is mapped onto the user-data parameter of the RT-OPEN response primitive.

In the case of error the responder of the association supplies the result parameter of the RT-OPEN response primitive with the "rejected (permanent)" or "rejected (transient)". In the case of "rejected (permanent)", the user-data parameter of the RT-OPEN response primitive shall be either authentication-error or unacceptable-dialogue-mode.

13.1.1.1.3 *Mode*

This parameter shall be supplied by the initiator of the association in the RT-OPEN request primitive, and shall have the value "X.410-1984 mode".

13.1.1.2 *MTA-unbind onto RT-CLOSE*

The MTA-unbind is mapped onto the RT-CLOSE service of the RTSE. In the X.410-1984 mode, the RT-CLOSE service has no parameters.

13.1.2 *Mapping onto RT-TRANSFER*

The message-transfer, probe-transfer and report-transfer services are mapped onto the RT-TRANSFER service of the RTSE.

An MTSE may issue an RT-TRANSFER request primitive only if it possesses the turn (see § 13.1.3) and if there is no outstanding RT-TRANSFER confirm primitive.

The use of the parameters of the RT-TRANSFER service is qualified in the following paragraphs.

13.1.2.1 *APDU*

The value of the MTS-APDU shall be mapped onto the APDU parameter of the RT-TRANSFER request primitive by the sender.

For the message-transfer service, the MTS-APDU is a message. For the probe-transfer service, the MTS-APDU is a probe. For the report-transfer service, the MTS-APDU is a report.

13.1.2.2 *Transfer-time*

The value of this parameter is specified by a local rule of the sender. It may be related to the priority of the APDU (see § 13.1.3.1.1).

13.1.3 *Managing the turn*

This paragraph describes managing the turn using the RT-TURN-PLEASE and RT-TURN-GIVE services of the RTSE.

The MTSE must possess the turn before it can use the RT-TRANSFER service to transfer a message, probe or report.

The MTSE without the turn may issue an RT-TURN-PLEASE request primitive, the priority parameter of which reflects the highest priority APDU awaiting transfer.

The MTSE with the turn may issue an RT-TURN-GIVE request primitive when it has no further APDUs to transfer. It shall issue an RT-TURN-GIVE request primitive in response to an RT-TURN-PLEASE indication primitive when it has no further APDUs to transfer of priority equal to, or higher than, that indicated in the RT-TURN-PLEASE indication primitive. If it has APDUs of lower priority still to transfer, it may then issue an RT-TURN-PLEASE request primitive, the priority parameter of which reflects the highest priority APDU awaiting transfer.

13.1.3.1 *Use of the RT-TURN-PLEASE service*

An MTSE issues the RT-TURN-PLEASE request primitive to request the turn. It may do so only if it does not already possess the turn.

If the initiator of the association supplied a dialogue-mode parameter value of "monologue" and an initial-turn parameter value of "association-initiator", the RT-TURN-PLEASE service shall not be used.

The use of the parameter of the RT-TURN-PLEASE service is qualified in the following paragraph.

13.1.3.1.1 *Priority*

The value of the priority parameter is supplied by the MTSE requesting the turn, and reflects the highest priority APDU awaiting transfer.

Priority zero is the highest priority, and is reserved for the action of releasing the asociation by the initiator.

Priority one shall be assigned to messages whose priority field (defined in § 8.2.1.1.1.8 of Recommendation X.411) has the value urgent. Priority one shall also be assigned to probes and reports.

Priority two shall be assigned to messages whose **priority** field is **normal**.

Priority three shall be assigned to messages whose **priority** field is **non-urgent**.

If more than one association is established between two MTAs, MTS-APDUs may be assigned to associations in accordance with their priorities. Several associations may be used to carry MTS-APDUs of the same priority. On any one association, higher priority MTS-APDUs are sent before lower priority MTS-APDUs; MTS-APDUs of the same priority are sent "first-in-first-out".

13.1.3.2 *Use of the RT-TURN-GIVE service*

An MTSE issues the RT-TURN-GIVE request primitive to relinquish the turn to its peer. It may do so only if it possesses the turn.

If the initiator of the association supplied a Dialogue-mode parameter value of "monologue" and an Initial-turn parameter value of "association-initiator", the RT-TURN-GIVE service shall not be used.

The RT-TURN-GIVE service has no parameters.

13.1.4 *Use of the RT-P-ABORT service*

The application-process is the user of the RT-P-ABORT service of the RTSE.

The RT-P-ABORT service provides an indication to the application-process that the application-association cannot be maintained (e.g., because recovery not possible).

The RT-P-ABORT service has no parameters.

13.1.5 *Use of the RT-U-ABORT service*

The RT-U-ABORT service of the RTSE is not available in X.410-1984 mode.

13.2 *Mapping onto RTSE normal mode*

This paragraph defines the mapping of the MTS transfer protocol (P1) onto used services for application-contexts that include the RTSE in normal mode. Support for this mapping is optional for conformance to this Recommendation. Note that ISO 10021-6, support for the RTSE in normal mode is mandatory.

Paragraph 13.2.1 defines the mapping of the MTA-bind and MTA-unbind services onto the RT-OPEN and RT-CLOSE services of the RTSE in normal mode. Paragraph 13.2.2 defines the mapping of the message-transfer, probe-transfer and report-transfer services onto the RT-TRANSFER service of the RTSE. Paragraph 13.2.3 describes managing the turn using the RT-TURN-PLEASE and RT-TURN-GIVE services of the RTSE. Paragraph 13.2.4 defines the use of the RT-P-ABORT service of the RTSE. Paragraph 13.2.5 defines the use of the RT-U-ABORT service of the RTSE.

13.2.1 *Mapping onto RT-OPEN and RT-CLOSE*

This paragraph defines the mapping of the MTA-bine and MTA-unbind services onto the RT-OPEN and RT-CLOSE services of the RTSE in normal mode.

13.2.1.1 *MTA-bind onto RT-OPEN*

The MTA-bind service is mapped onto the RT-OPEN service of the RTSE. The use of the parameters of the RT-OPEN service is qualified in the following paragraphs.

13.2.1.1.1 *Mode*

This parameter shall be supplied by the initiator of the association in the RT-OPEN request primitive, and shall have the value "normal mode".

13.2.1.1.2 *Application context name*

The initiator of the association shall propose the **mts-transfer** application-context defined in this Recommendation in the RT-OPEN request primitive.

13.2.1.1.3 *User-data*

The mapping of the bind-operation of the MTA-bind service onto the user-data parameter of the RT-OPEN request primitive is defined in Recommendation X.219.

13.2.1.1.4 *Presentation context definition list*

The initiator of the association supplies the presentation context definition list in the RT-OPEN request primitive.

The presentation context definition list comprises a presentation-context-definition for each abstract-syntax included in the application-context. A presentation-context-definition comprises a presentation-context-identifier and an abstract-syntax-name for the ASE. The named abstract-syntax for the RTSE includes the abstract-syntax for the bind-operation.

Paragraph 12 defines the abstract-syntaxes included in the application-context.

13.2.1.2 *MTA-unbind onto RT-CLOSE*

The MTA-unbind is maped onto the RT-CLOSE service of the RTSE.

No parameters of the RT-CLOSE service are used in normal mode.

13.2.2 *Mapping onto RT-TRANSFER*

The message-transfer, probe-transfer and report-transfer services are mapped onto the RT-TRANSFER service of the RTSE.

The mapping of these services onto the RT-TRANSFER service in normal mode is identical to the mapping in X.410-1984 mode, defined in § 13.1.2.

13.2.3 *Managing the turn*

The MTSE must possess the turn before it can use the RT-TRANSFER service to transfer a message, probe or report.

Managing the turn in normal mode is identical to managing the turn in X.410-1984 mode, defined in § 13.1.3.

13.2.4 *Use of the RT-P-ABORT service*

The application-process is the user of the RT-P-ABORT service of the RTSE.

The RT-P-ABORT service provides an indication to the application-process that the application-association cannot be maintained (e.g. because recovery not possible).

The RT-P-ABORT service has no parameters.

Note that the use of the RT-P-ABORT service in normal mode is identical to the use of the RT-P-ABORT service in X.410-1984 mode.

13.2.5 *Use of the RT-U-ABORT service*

The application-process is the user of the RT-U-ABORT service of the RTSE.

The RT-U-ABORT service enables the application-process to abort the application-association. The RT-U-ABORT service may be requested by either the initiator or the responder of the association.

No parameters of the RT-U-ABORT service are used in normal mode.

Note that the RT-U-ABORT service is not available in X.410-1984 mode.

14 Conformance

An MD claiming conformance to the MTS transfer protocol (P1) specified in this Recommendation shall comply with the requirements in §§ 14.1, 14.2 and 14.3.

14.1 *Statement requirements*

The following shall be stated:

a) the application-contexts defined in Section 3 of this Recommendation for which conformance is claimed;

b) whether monologue, two-way alternate, or both monologue and two-way alternate dialogue-modes are supported;

c) whether the MD can act as the initiator, or the responder, or either the initiator or the responder, of an association.

Table 6/X.419 classifies the support for application-contexts required for conformance to the MTS transfer protocol (P1).

TABLE 6/X.419

MTS transfer protocol conformance requirements

Application context	MD
MTS transfer protocol	
mts-transfer-protocol-1984	Mandatory
mts-transfer-protocol	Mandatory
mts-transfer	Optional

14.2 *Static requirements*

The MD shall:

a) conform to the abstract-syntax definition of the MTS transfer protocol (P1) defined in § 12 of this Recommendation.

14.3 *Dynamic requirements*

The MD shall:

a) conform to the procedures for distributed operation of the MTS defined in Recommendation X.411;

b) conform to the mapping onto used services defined in § 13 of this Recommendation, required by the application-contexts for which conformance is claimed; support for the mapping onto the RTSE in X.410-1984 mode is mandatory, and support for the mapping onto the RTSE in normal mode is optional;

c) conform to the rules for interworking with MDs conforming to Recommendation X.411 (1984) defined in Annex B of this Recommendation;

d) conform to the use of underlying services defined in § 11.3 of this Recommendation.

ANNEX A

(to Recommendation X.419)

Reference definition of MHS protocol object identifiers

This Annex defines for reference purposes various object identifiers cited in the ASN.1 modules in the body of this Recommendation. The object identifiers are assigned in Figure 6/X.419.

All object identifiers that this Recommendation assigns are assigned in this Annex. However, this Annex is not definitive for all assignments. Other definitive assignments occur in the modules in the body of this Recommendation and are referred to in this Annex.

```
MHSProtocolObjectIdentifiers { joint-iso-ccitt mhs-motis(6) protocols(0) modules(0) object-identifiers(0) }

DEFINITIONS IMPLICIT TAGS ::=

BEGIN


-- Prologue
-- Exports everything

IMPORTS -- nothing -- ;


-- MHS protocols

id-mhs-protocols OBJECT IDENTIFIER ::= { joint-iso-ccitt mhs-motis(6) protocols(0) } -- not definitive


-- Categories of object identifiers

id-mod OBJECT IDENTIFIER ::= { id-mhs-protocols 0 }                -- modules

id-ac OBJECT IDENTIFIER ::= { id-mhs-protocols 1 }                 -- application contexts

id-as OBJECT IDENTIFIER ::= { id-mhs-protocols 2 }                 -- abstract syntaxes

id-ase OBJECT IDENTIFIER ::= { id-mhs-protocols 3 }                -- application service elements


-- Modules

id-mod-object-identifiers OBJECT IDENTIFIER ::= { id-mod 0 }       -- not definitive

id-mod-mts-access-protocol OBJECT IDENTIFIER ::= { id-mod 1 }      -- not definitive

id-mod-ms-access-protocol OBJECT IDENTIFIER::= { id-mod 2 }        -- not definitive

id-mod-mts-transfer-protocol OBJECT IDENTIFIER ::= { id-mod 3 }    -- not definitive
```

Figure 6/X.419 (Part 1 of 3)

Abstract syntax definition of MHS protocol object identifiers

```
-- Application contexts
-- MTS access protocol
id-ac-mts-access OBJECT IDENTIFIER ::= { id-ac 0 }
id-ac-mts-forced-access OBJECT IDENTIFIER ::= { id-ac 1 }
id-ac-mts-reliable-access OBJECT IDENTIFIER ::= { id-ac 2 }
id-ac-mts-forced-reliable-access OBJECT IDENTIFIER ::= { id-ac 3 }

-- MS access protocol
id-ac-ms-access OBJECT IDENTIFIER ::= { id-ac 4 }
id-ac-ms-reliable-access OBJECT IDENTIFIER ::= { id-ac 5 }

--MTS transfer protocol
id-ac-mts-transfer OBJECT IDENTIFIER ::= { id-ac 6 }

-- Abstract syntaxes
id-as-acse OBJECT IDENTIFIER ::= { joint-iso-ccitt association-control (2) abstract-syntax (1) opdus (0) version1 (1) }
id-as-msse OBJECT IDENTIFIER ::= { id-as 1 }
id-as-mdse OBJECT IDENTIFIER ::= { id-as 2 }
id-as-mrse OBJECT IDENTIFIER ::= { id-as 5 }
id-as-mase OBJECT IDENTIFIER ::= { id-as 6 }
id-as-mtse OBJECT IDENTIFIER ::= { id-as 7 }
id-as-mts-rtse OBJECT IDENTIFIER ::= { id-as 8 }
id-as-ms OBJECT IDENTIFIER ::= { id-as 9 )
```

Figure 6/X.419 (Part 2 of 3)

Abstract syntax definition of MHS protocol object identifiers

```
id-as-ms-rtse OBJECT IDENTIFIER ::= { id-as 10 }
id-as-mts OBJECT IDENTIFIER ::= { id-as 11 }

-- Application service elements
id-ase-msse OBJECT IDENTIFIER ::= { id-ase 0 }
id-ase-mdse OBJECT IDENTIFIER ::= { id-ase 1 }
id-ase-mrse OBJECT IDENTIFIER ::= { id-ase 2 }
id-ase-mase OBJECT IDENTIFIER ::= { id-ase 3 }
id-ase-mtse OBJECT IDENTIFIER ::= { id-ase 4 }

END    -- of MHSProtocolObjectIdentifiers
```

Figure 6/X.419 (Part 3 of 3)

Abstract syntax definition of MHS protocol object identifiers

ANNEX B

(to Recommendation X.419)

Interworking with 1984 systems

This Annex defines the rules to be obeyed by MDs claiming conformance to this Recommendation (hereafter referred to as "1988 systems") when interworking with implementations conforming to Recommendation X.411 (1984) (hereafter referred to as "1984 systems") using the MTS transfer protocol (P1).

Paragraph B.1 defines the rules for establishing associations that a 1988 system shall obey when interworking with a 1984 system.

Paragraph B.2 defines the rules that a 1988 system shall obey when transferring an MTS-APDU to a 1984 system.

Paragraph B.3 defines the rules that a 1988 system shall obey when receiving an MTS-APDU from a 1984 system.

Note – As Recommendation X.411 (1984) only defines the interactions at the boundary of an ADMD, the interworking rules in this Annex only apply at such a boundary.

Additional types have been added to the universal class of ASN.1 types compared to those defined in Recommendation X.409 (1984). The valid replacement specifications for an ANY type are therefore extended. Note that 1984 systems may be unable to handle the extended universal types. It is likely that a 1984 system may correctly handle these fields even if they contain the extended types. However, such fields intended for a 1984 system should be restricted to the universal types defined in Recommendation X.409 (1984).

The basic encoding rules for ASN.1 give more flexibility than Recommendation X.409 (1984) for the long form of the length octets. The former permits the use of more length octets than the minimum necessary, whereas the latter does not. Therefore, when interworking with a 1984 system, it is necessary to obey this restriction, and use the fewest possible number of octets, with no leading octets having the value 0.

B.1 *Association establishment*

This paragraph defines the restrictions that a 1988 system shall observe with the MTA-bind when establishing an association with a 1984 system. There are no restrictions with the MTA-unbind.

The **mts-transfer-protocol-1984**, as defined in § 12, shall be used for compatibility with the 1984 system.

B.1.1 *Initiator-credentials/responder-credentials*

There are no restrictions placed on these elements as the corresponding elements in Recommendation X.411 (1984) were each defined to be ANY type. Note, however, that a 1984 system will be restricted in its use of these elements when interworking with 1988 systems as described above.

B.1.2 *Security-context*

This optional element shall not be generated by a 1988 system when interworking with a 1984 system. Note that a 1984 system is not capable of generating this element.

B.1.3 *Bind-error*

The bind-error value **unacceptable-security-context** shall not be generated by a 1988 system.

B.2 *Rules for transferring to 1984 systems*

This paragraph defines the interworking rules that a 1988 system shall obey when transferring an MTS-APDU to a 1984 system. The transformation of an MTS-APDU conforming to Recommendation X.411 to one conforming to Recommendation X.411 (1984) is called *downgrading.* The rules are expressed in terms of the actions to be taken on each protocol element of the MTS transfer protocol (P1) by the 1988 system.

For a given MTS-APDU, if none of the rules deem that downgrading would fail, then the MTS-APDU shall be downgraded in accordance with all applicable rules before being transferred to the 1984 system.

If one or more of the rules deem that downgrading has failed, then the action taken by the MTA is the same as if the transfer had failed (see § 14 of Recommendation X.411).

Note – The potential or actual loss of information caused by applying these rules may affect an MTA's routing strategy.

The remainder of this paragraph specifies the rules for each of the protocol elements. Protocol elements not specifically mentioned shall be transferred unchanged. Unless otherwise specified, the rules specified apply in whichever MTS-APDU the protocol elements appear.

B.2.1 *Extensions*

If any per-message **extensions** elements are present, and no **extension-field** is marked **critical-for-transfer** or **critical-for-delivery**, the **extensions** elements shall be deleted.

If any per-message **extensions** elements are present, and any **extension-field** is marked **critical-for-transfer** or **critical-for-delivery**, downgrading shall fail.

These rules shall be applied before any of the rules described in the following paragraphs.

B.2.2 *Per-domain-bilateral-information*

If a **private-domain-identifier** is present in an element of **per-domain-bilateral-information**, then that element of **per-domain-bilateral-information** shall be deleted.

Otherwise, the **per-domain-bilateral-information** shall be unchanged.

B.2.3 *Trace-information/subject-intermediate-trace-information*

If an **other-actions** element is present in any **trace-information-elements** or **subject-intermediate-trace-information-elements**, the **other-actions** element shall be deleted.

Otherwise, the **trace-information** or **subject-intermediate-trace-information** shall be unchanged.

B.2.4 *Originator-name/report-destination-name*

If the **originator-name** in a **message-transfer-envelope** or a **probe-transfer-envelope**, or if the **report-destination-name** in a **report-transfer-envelope**, cannot be downgraded according to the rules given for **OR-name** (see § B.2.7), then downgrading shall fail.

Otherwise the element shall be unchanged.

B.2.5 *Per-recipient-fields of message- or probe-transfer*

If a **recipient-name** in the **per-receipient-fields** of a **message-transfer-envelope** or a **probe-transfer-envelope** cannot be downgraded according to the rules given for **OR-name** (see § B.2.7), or any **per-recipient extension-field** exists and is marked **critical-for-transfer** or **critical-for-delivery**, then:

a) if the corresponding **responsibility** element has the value **responsible**, then downgrading shall fail;

b) if the corresponding **responsibility** element has the value **not-responsible**, the the element for that recipient shall be deleted from **per-recipient-fields.**

Note – The downgrading rules imply that **disclosure-of-recipients** is neither critical-for-transfer nor critical-for-delivery.

B.2.6 *Per-recipient-fields of report-transfer*

If an **actual-recipient-name** or an **intended-recipient-name** in the **per-recipient-fields** of a **report-transfer-content** cannot be downgraded according to the rules given for **OR-name** (see § B.2.7), then the corresponding element of **per-recipient-fields** shall be deleted. If all the elements of **per-recipient-fields** are so deleted, downgrading shall fail.

B.2.7 *OR-name*

The **OR-name** shall be downgraded by deleting the **directory-name**, if present, and by downgrading the **OR-address** (see § B.2.8).

B.2.8 *OR-address*

If the **OR-address** contains any attributes encoded both as teletext strings and as printable strings, the teletex strings shall be deleted.

If the **OR-address** is a **numeric-OR-address** or a **terminal-OR-address** containing a **private-domain-name**, the **OR-address** cannot be downgraded.

If the **OR-address** is a **telematic-OR-address**:

a) that contains a **country-name**, an **administration-domain-name**, a **network-address**, optionally **domain-defined-attributes**, and no others, the **OR-address** shall be unchanged;

b) that contains a **network-address**, optionally a **terminal-identifier**, and no others, the **OR-address** shall be unchanged;

c) that contains combinations of attributes other than the above, all attributes except the **network-address** and the **terminal identifier**, if present, shall be deleted.

If the **OR-address** contains any attributes encoded as teletex strings and the corresponding printable strings are absent, the **OR-address** cannot be downgraded.

If after applying all the above rules the **OR-address** still contains any **extension-attributes**, the **OR-address** cannot be downgraded.

B.2.9 *Encoded-information-types*

Basic **encoded-information-types** indicated by object identifiers shall be mapped to the corresponding bit in **basic-encoded-information-types**, and the object identifiers shall be deleted.

Other **encoded-information-types** indicated by object identifiers shall be mapped to the **undefined** bit in **basic-encoded-information-types**, and the object identifiers shall be deleted.

Any **non-basic-parameters** other than for **g4-class-1** and **mixed-mode** types shall not be altered. Those for **g4-class-1** and **mixed-mode** may be transformed according to rules deduced from Recommendations T.73 (1984), T.400, T.501 and T.503; if this is not possible, downgrading shall fail.

Notwithstanding the above rules, **encoded-information-types** in a **report-transfer-content** shall be deleted.

B.2.10 *Content-type and content*

If the **content-type** in a message or probe is indicated by integer, it shall be unchanged. The **content** in the message shall also be unchanged.

If the **content-type** in a message is indicated by an object identifier, it shall be mapped to the integer value **external** in place of the object identifier. The object identifier and the **content** shall be combined together into a value of the EXTERNAL type, and this value shall be the contents of the new **content**. The object identifier shall be the EXTERNAL's direct-reference and the contents of the **content** OCTET STRING shall be its octet-aligned encoding. The encoding of the **content** OCTET STRING shall be the Basic Encoding Rules of ASN.1

If the **content-type** in a probe is indicated by an object identifier, downgrading shall fail.

The **content-type** in a report shall be deleted. The **returned-content** shall be unchanged.

B.3 *Rules for receiving from 1984 systems*

This paragraph defines the interworking rules which a 1988 system shall obey upon receiving an MTS-APDU from a 1984 system.

Size constraints have been defined for a number of MTS transfer protocol (P1) elements. Providing that a 1984 system observes these constraints, a correctly encoded MTS-APDU received from a 1984 sytem also conforms to 1988 MTS protocol (P1). Therefore, a 1988 system need take no special action.

B.4 *Service irregularities*

The use of redirection and distribution lists in the presence of 1988/1984 domain boundaries may lead to some irregularities which are listed below:

- recipients may not be able to notice that they received a message because of DL expansion or redirection;
- when a message traverses a 1984 domain, the expansion history and the redirection history are lost. This mat cause premature routing hop detection and result in redirection or expansion failure. Note that only a DL with a 1984 compatible O/R address may encounter this problem;
- 1984 MTAs will return notifications to the message originator rather than redirecting them back along the DL expansion path;
- 1984 systems may see new distinguished values for integer protocol elements which are unknown to them.

ANNEX C

(to Recommendation X.419)

Differences between 1984 and 1988 MHS protocols

This Annex identifies the differences between the MTS access protocol (P3) and MTS transfer protocol (P1) defined in this Recommendation and the P3 and P1 protocols defined in Recommendation X.411 (1984). Differences of a purely editorial nature are not included here.

The differences are identified in terms of the additions or other changes made to protocol elements present in P3 and P1 are defined in Recommendation X.411 (1984). The differences are more precisely indicated in the abstract syntax definitions in Recommendation X.411, in which every data type that has been changed is highlighted by means of underlining.

Paragraph C.1 identifies the differences in the MTS access protocol (P3). Paragraph C.2 identifies the additional differences in the MTS transfer protocol (P1).

C.1 *MTS access protocol (P3) differences*

This paragraph identifies the differences between the MTS access protocol (P3) defined in this Recommendation and the P3 protocol defined in Recommendation X.411 (1984).

C.1.1 *Size constraints*

Constraints to limit the length of string types, the number of items in a SET OF or SEQUENCE OF type, and the value range of INTEGER types have been placed on all parameters defined in Recommendation X.411 (1984) with the exception of the message **content**.

C.1.2 *Changes to fundamental types*

The parameters **OR-name**, **content-type**, **encoded-information-types** and **content**, which occur in various places in the operation arguments and results, have been extended, as described below.

C.1.2.1 *OR-name*

Two new optional parameters have been added to **OR-name**.

The first of these is a set of **extension-attributes** that provide the means of using the teletex character set for the **standard-** and **domain-defined-attributes**, of specifying a **postal-OR-address** for physical delivery, and of specifying a **terminal-address** from an **extended-network-address**.

The second of these is a **directory-name**, as defined in Recommendation X.501.

If only **standard-, domain-defined-** or **extension-attributes** are present, then the **OR-name** constitutes an **OR-address**. Otherwise, a **directory-name** is also present. If a **directory-name** alone is present, it may be necessary to map the **directory-name** to an **OR-address** (e.g., using the directory).

C.1.2.2 *Content-type*

The option of identifying the **content-type** with an object ifentifier instead of an integer has been added. It is the preferred method of identifying new **content-types**, and the assignment of new integer values is discouraged. Three new values have been defined for the integer choice: **undefined, external** and **interpersonal-messaging-1988**.

C.1.2.3 *Encoded-information-types*

The option of specifying a set of external **encoded-information-types** has been added. All new **encoded-information-types** will be added as an object identified.

The definition of the **non-basic-parameters** for the **g4-class-1** and **mixed-mode** types has been amended in that the definition referenced in Recommendations T.400, T.501 and T.503 has changed from that previously referenced in Recommendation T.73 (1984), and in that it now uses explicit instead of implicit tagging.

C.1.2.4 *Content*

The **content** of a message is still of type OCTET STRING. If the **content-type** is identified by the integer value **external**, the **content** is termed an **external-content**. The value of the OCTET STRING for an **external-content** shall be the ASN.1 encoding of an EXTERNAL.

C.1.3 *Extensions*

Most of the extensions to the MTS abstract service definied in Recommendation X.411 are accomodated in the protocol by the addition of a single new parameter **extensions** into the operation envelopes and results. The parameter is absent when no extensions are required. It may be present in the:

- **Message-submission-envelope**, on a per-message and per-recipient basis;
- **Message-submission-result**;
- **Probe-submission-envelope**, on a per-probe and per-recipient basis;
- **Probe-submission-result**;
- **Message-delivery-envelope**; and
- **Report-delivery-envelope**, on a per-report and per-recipient basis.

C.1.4 *Bind*

In Recommendation X.411 (1984), credentials of type ANY are exchanged using the bind argument and result. The type of the ANY is restricted in this Recommendation to a choice of **simple-credentials** (either an IA5String or an OCTET STRING), or **strong-credentials** based on cryptographic techniques.

An optional parameter to specify a **security-context** has been added to the argument. A new error has been added to indicate an **unacceptable-security-context**.

C.1.5 *Message-submission*

The **original-encoded-information-types** and **explicit-conversion** parameters in the **message-submission-envelope** have been made optional.

Two new errors have been added: **inconsistent-request** and **security-error**.

C.1.6 *Probe-submission*

As for message-submission, see § C.1.5.

C.1.7 *Cancel-deferred-delivery*

This operation is virtually unchanged with the exception of the size constraints described in § C.1.1 and the removal of the message transferred error (subsumed by deferred-delivery-cancellation-rejected).

C.1.8 *Submission-control*

An optional parameter **permissible-security-context** has been added to the argument.

An optional parameter **waiting-context-types** has been added to the result to specify the **content-types** of any waiting messages held due to prevailing controls. The indicator **other-security-labels** has been added to the **waiting-messages** parameter of the result.

An error has been added: **security-error**.

C.1.9 *Message-delivery*

The **original-encoded-information-types** and **delivery-flags** parameters have been made optional in the **message-delivery-envelope**, and an optional parameter **content-identifier** has been added to it.

The operation has been made confirmed by adding a RESULT clause, which contains two optional security parameters: **recipient-certificate** and **proof-of-delivery**.

One new error has been added: **security-error**.

C.1.10 *Report-delivery*

Two new optional parameters have been added to the **report-delivery-envelope**: the **content-type** and the **original-encoded-information-types** of the original message.

Five new **non-delivery-reason-codes** and 35 new **non-delivery-diagnostic-codes** have been defined.

Five new values of the **type-of-MTS-user** parameter havebeen added: **message-store**, **distribution-list**, **physical-delivery-access-unit**, **physical-recipient** and **other**.

The operation has been made confirmed by adding a RESULT clause (which conveys no parameters).

One new error has been added: **security-error**.

C.1.11 *Delivery-control*

Two new optional control parameters have been added to the argument: **permissible-content-types** and **permissible-security-context**.

An optional **waiting-content-types** parameter has been added to the result.

Two new errors have been added: **control-violates-registration** and **security-error**.

C.1.12 *Register*

Two new optional parameters have been added to the argument: **deliverable-content-types** and **labels-and-redirections**.

The tags on the restrict, **permissible-operations** and **permissible-maximum-content-length** parameters of the **default-delivery-controls** have been altered. The **permissible-content-types** parameter has been added.

C.1.13 *Change-credentials*

This possible types supplied fot the credentials in this operation have been restricted, as described in § C.1.4. The relationship between the types supplied for the **old-credentials** and **new-credentials** has also been restricted (to be of the same type).

C.2 *MTS transfer protocol (P1) differences*

This paragraph identifies the differences between the MTS transfer protocol (P1) defined in this Recommendation and the P1 protocol defined in Recommendation X.411 (1984).

The following changes to the MTS transfer protocol (P1) are the same as those defined for the MTS access protocol (P3): size constraints (see § C.1.1), changes to fundamental types (see § C.1.2) and bind (see § C.1.4).

The following paragraphs detail other changes to the MTS transfer protocol (P1).

C.2.1 *External-fields*

The new parameter **extensions** is used to include most of the abstract-service extensions to the MTS transfer protocol (P1) (see § C.1.3). The parameter is absent when no extensions are required. It may be present in the:

- **Message-transfer-envelope**, on a per-message and per-recipient basis.
- **Probe-transfer-envelope**, on a per-probe and per-recipient basis.
- **Report-transfer-envelope.**
- **Report-transfer-content**, on a per-report and per-recipient basis.

C.2.2 *Others differences*

Two optional parameters have been added to the per-report transfer fields of the **report-transfer-envelope**: **original-encoded-information-types** and **content-type.**

An optional **private-domain-identifier** has been added to the **per-domain-bilateral-information** parameter of the message- and **probe-transfer-envelopes**. This permits **per-domain-bilateral-information** to be sent to PRMDs as well as ADMDs.

An optional **other-actions** parameter has been added to the elements of **trace-information.** The new parameter conveys two flags: **redirected** to indicate that the message was redirected by that MD, and **expanded** to indicate that the MD expanded a distribution-list.

ANNEX D

(to Recommendation X.419)

Differences between ISO and CCITT versions

This Annex identifies the technical differences between the ISO and CCITT versions of the text of CCITT Recommendations X.419 and ISO 10021-6 as they relate to the support of the MTS transfer protocol (P1).

They are:

1) In CCITT Recommendation X.419, it is a mandatory conformance requirement to have the capability to interwork with implementations of the CCITT Recommendation X.411 (1984) using the MTS Transfer Protocol (P1) (for ADMD - ADMD and ADMD - PRMD). In ISO 10021-6, the capability to interwork with 1984 systems is optional (for PRMD - PRMD and intra-domain).
2) In CCITT Recommendation X.419, support for the mapping of the MTS transfer protocol (P1) onto the RTSE in X.410-1984 mode is a mandatory conformance requirement; support for the mapping onto the RTSE in normal mode is optional. In ISO 10021-6, support for the mapping onto the RTSE in normal mode is mandatory, support for the mapping onto the RTSE in X.410-1984 mode is optional.

   Note – An implementation conformant only to the mandatory mapping of ISO 10021-6 would not be capable of interworking with implementations of the CCITT Recommendation X.411 (1984), nor implementations conformant only to the mandatory mapping of CCITT Recommendation X.419 (1988), and vice versa.
3) In CCITT Recommendation X.419, requirements are made for the support of lower layer services (see § 11.3.4). In ISO 10021-6, these requirements are omitted.

Recommendation X.420

MESSAGE HANDLING SYSTEMS: INTERPERSONAL MESSAGING SYSTEM [1]

(Malaga-Torremolinos, 1984; amended at Melbourne, 1988)

The establishment in various countries of telematic services and computer-based store-and-forward message services in association with public data networks creates a need to produce standards to facilitate international message exchange between subscribers to such services.

The CCITT,

considering

(a) the need for message handling systems;

(b) the need to transfer and store messages of different types;

(c) that Recommendation X.200 defines the reference model of open systems interconnection for CCITT applications;

(d) that Recommendations X.208, X.217, X.218 and X.219 provide the foundation for CCITT applications;

(e) that the X.500-series Recommendations define directory systems;

(f) that message handling systems are defined in a series of Recommendations X.400, X.402, X.403, X.407, X.408, X.411, X.413, and X.419;

(g) that interpersonal messaging is defined in Recommendations X.420 and T.330,

unanimously declares

(1) that the abstract information objects users exchange in interpersonal messaging are defined in Section 2;

(2) that the abstract service offered to users in interpersonal messaging is defined in Section 3;

(3) that how that abstract service is provided is specified in Section 4.

TABLE OF CONTENTS

[1] Recommendation X.420 and ISO 10021-7 [Information Processing Systems – Text Communication – MOTIS – Interpersonal Messaging System] were developed in close collaboration and are technically aligned, except for the differences.

SECTION 1 – INTRODUCTION

0 Introduction

This Recommendation is one of a set of Recommendations for message handling. The entire set provides a comprehensive blueprint for a message handling system (MHS) realized by any number of cooperation open systems.

The purpose of an MHS is to enable users to exchange messages on a store-and-forward basis. A message submitted on behalf of one user, the originator, is conveyed by the message transfer system (MTS) and subsequently delivered to the agents of one or more additional users, the recipients. Access units (AUs) link the MTS to communication systems of other kinds (e.g., postal systems). A user is assisted in the preparation, storage, and display of messages by a user agen (UA). Optionally, it is assisted in the storage of messages by a message store (MS). The MTS comprises a number of message transfer agents (MTAs) which collectively perform the store-and-forward message transfer function.

This Recommendation defines the message handling application called *interpersonal messaging*, specifying in the process the message content type and associated procedures known as *P2*.

The text of this Recommendation is the subject of joint CCITT-ISO agreement. The corresponding ISO/IEC specification is ISO 10021-7.

1 Scope

This Recommendation defines **interpersonal messaging**, a form of message handling tailored for ordinary interpersonal business or private correspondence.

This Recommendation is one of a series on message handling. Recommendation X.402 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. Recommendation X.402 identifies those documents as well.

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

The requirements for conformance to this Recommendation are given in § 22.

2 References

This Recommendation cites Recommendation X.402, many of the documents it cites, and those below.

ISO Standard 639.2	Code for the representation of names of languages.
Recommendation T.4	Standardization of Group 3 facsimile apparatus for document transmission.
Recommendation T.30	Procedures for document facsimile transmission in the general switched telephone network.
Recommendation T.100	International information exchange for interactive videotex.
Recommendation T.101	International interworking for videotex services.
Recommendation T.330	Telematic access to IPMS.
Recommendation X.420 (1984)	Message handling systems: Interpersonal messaging user agent layer.
	X.400-series implementor's guide, Version 6, 6 November 1987.

3 Definitions

For the purposes of this Recommendation, the definitions of Recommendation X.402 apply.

4 Abbreviations

For the purposes of this Recommendation, the abbreviations of Recommendation X.402 apply.

5 Conventions

This Recommendation uses the descriptive conventions identified below.

5.1 *ASN.1*

This Recommendation 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;

b) to define the functional objects of interpersonal messaging, the OBJECT and REFINE macros of Recommendation X.407;

c) to define the abstract service of interpersonal messaging, the PORT and ABSTRACT-OPERATION and -ERROR macros of Recommendation X.407;

d) to define the *heading extensions*, the HEADING-EXTENSION macro of § 7.2.17;

e) to define *extended body part types*, the EXTENDED-BODY-PART-TYPE macro of § 7.3.12;

f) to define MS attributes, the ATTRIBUTE macro of Recommendation X.500.

The various uses of the ASN.1 notation are summarized in Table 1/X.420. With the two exceptions evident from the table, whenever ASN.1 is employed, it appears both in the body of the Recommendation to aid the exposition, and again, largely redundantly, in an Annex for reference.

TABLE 1/X.420

Uses of the ASN.1 notation

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

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

Note that ASN.1 tags are implicit throughout the ASN.1 module the Annex defines; the module is definitive in that respect.

Note 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.

Note 2 – The use of the ABSTRACT-OPERATION and -ERROR macros, derived from the correspondingly named macros 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 macros and with minimal additional specification, actually could be invoked via ROS is immaterial in the present context.

5.2 *Grade*

This Recommendation uses the concept of grade as developed in Recommendation X.402.

5.3 *Terms*

Throughout this Recommendation, 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.

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.

Note 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 § 20). These words, therefore, refer to the roles users and DLs play in such transmittals.

Note 2 – An *IPM* may appear (see § 7.3.8) 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.

Note 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:

a) **heading**: A set of **heading fields** (or **fields**), each an information item that gives a characteristic of the IPM (e.g., its importance);

b) **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/X.420.

This paragraph 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.

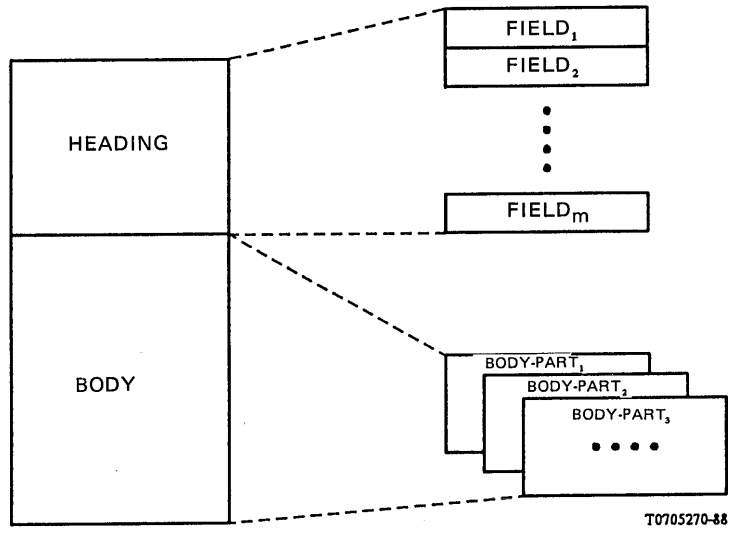


FIGURE 1/X.420

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*, and *O/R descriptor*-- 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                        ORAddress 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 O/R addresses. 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 K). 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 Recommendation assigns.

7.1.2 *Recipient specifier*

A **recipient specifier** is an information item that identifies a (preferred) 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 }
```

A recipient specifier has the following components:

a) **recipient** (M): Identifies the preferred recipient in question. An *O/R descriptor.*

If the *notification-requests* or *reply-requested* component makes a request of the preferred recipient, the *formal-name* component of the *O/R descriptor* above shall be present.

b) **notification-requests** (D no values): May make certain requests of the preferred 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 tha 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 § 8.

ii) *nrn:* A *non-receipt notification* is requested in the circumstances prescribed in § 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 preferred 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 preferred recipients. Furthermore, a user of whom a reply is requested may refrain from replying.

7.1.3 *O/R descriptor*

An **O/R 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 }
```

An O/R descriptor has the following components:

a) **formal-name** (C): Identifies the user or DL in question. One of its O/R 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 O/R descriptor appears in the *reply recipients* heading field.

iii) The O/R descriptor is the recipient component of a recipient specifier and the conditions stated in item 2 of § 7.1.2 are satisfied.

b) **free-form-name** (O): Identifies the user or DL in question. A telex string of from zero to a prescribed number of characters (see Annex K), 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 K), chosen from the graphical subset of the printable string character set. A length of zero is discouraged.

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

Note – One or more O/R 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, and O/R descriptor may appear in the following notification fields (see § 8): IPN originator and IPM preferred recipient.

7.2 *Heading fields*

The fields that 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 {},
        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 O/R 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 O/R 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 Recommendation; it is given meaning by users.

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

Note – Suppose, for example, 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 not 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 Recommendation; 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 Recommendation; it is given meaning by users.

Note – The copy recipients, for example, 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 *Obsoleted IPMs*

The **obsoleted 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.

ObsoletedIPMsField ::= SEQUENCE OF ObsoletedIPMsSubfield

ObsoletedIPMsSubfield ::= 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 Recommendation; it is given meaning by users.

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

Note 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 K), chosen from the graphic subset of the teletex string character set. A length of zero is discouraged.

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

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 preferred recipients of any replies to the present IPM. It comprises a sequence of sub-fields, each an O/R descriptor, one for each user or DL.

ReplyRecipientsField ::= SEQUENCE OF ReplyRecipientsSubfield

ReplyRecipientsSubfield ::= ORDescriptor

This conditional field shall be present if, and only 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 Recommendation; 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 preferred recipients as individuals, rather than in their professional capacities.

b) *private:* The IPM should be conveyed to no one other than its preferred recipients.

c) *company-confidential:* The IPM contains information that should be handled according to company-specific procedures.

The 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 **heading extensions** (or **extensions**), each conveying one such information item.

```
ExtensionsField ::= SET OF HeadingExtension

HeadingExtension ::= SEQUENCE {
        type        OBJECT IDENTIFIER,
        value       ANY DEFINED BY 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 abstrct 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.

All extensions are defined in Annex A. Thus, each extension's type component shall have one of the values given in that Annex. An extension whose type component has another value shall be ignored.

Every extension is defined by means of the following macro.

```
HEADING-EXTENSION MACRO ::=
BEGIN
        TYPE   NOTATION ::= "VALUE" type | empty
        VALUE  NOTATION ::= value (VALUE OBJECT IDENTIFIER)

END
```

An instance of the macro's type notation identifies the data type to which the extension's value component shall be restricted. If no type is identified explicitly, null is implied.

An instance of the macro's value notation identifies the object identifier that shall appear as the extension's type component.

Note – Future versions of this Recommendation may define additional extensions. Furthermore, future versions are likely to add to the heading only by means of this field.

7.3 *Body part types*

The types of body parts that may appear in the body of an IPM are defined and described below:

```
BodyPart ::= CHOICE {
        ia5-text                [0]  IA5TextBodyPart,
        voice                   [2]  VoiceBodyPart,
        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,
        externally-defined      [15] ExternallyDefinedBodyPart }
```

Body parts of some of the types defined below have two components, *parameters* and *data.* The **parameters** component (M) comprises a sequence of information items that describe the information object the body part represents and that typically are format and control parameters. The **data** component (M) is the information object itself.

Note 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. These tags, therefore, are avoided in BodyPart.

Note 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.

7.3.1 *IA5 text*

An **IA5 text** body part represents IA5 text. It has parameters and data components.

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

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

IA5TextData ::= IA5String
```

The parameters component comprises the following parameters:

a) **repertoire** (D *IA5*): Identifies the character set to which the data component is constrained.

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

This parameter may assume one of the following values:

i) *ITA2:* The data component shall be limited to 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).

7.3.2 *Voice*

A **voice** body part represents speech. It has parameter and data components.

```
VoiceBodyPart ::= SEQUENCE {
        parameters              VoiceParameters,
        data                    VoiceData }

VoiceParameters ::= SET -- for further study

VoiceData ::= BIT STRING -- for further study
```

The parameters of such a body part, and the digitized speech encoding technique that those parameters might identify and parameterize, are for further study.

The data component is the speech, a bit string.

7.3.3 *G3 facsimile*

A **G3 facsimile** body part represents Group 3 facsimile images. It has parameters and data components.

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

G3FacsimileParameters ::= SET {
        number-of-pages         [0]  INTEGER OPTIONAL,
        non-basic-parameters    [1]  G3FascimileNonBasicParameters 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 shall be present if (but not only if) the body contains two or more G3 facsimile body parts.

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 Recommendations T.4 and T.30.

Note 1 – The number-of-pages component identifies the number of elements in the sequence that constitutes the data component and is thus redundant.

Note 2 – 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.

7.3.4 *G4 Class 1*

A **G4 Class 1** body part represents a final-form of document of the sort that is processable by Group 4 Class 1 facsimile terminals. It comprises a sequence of protocol elements which describe the document's layout structure.

```
G4Class1BodyPart ::= SEQUENCE OF ProtocolElement
```

7.3.5 *Teletex*

A **teletex** body part represents a teletex document. It has parameters and data components.

```
TeletexBodyPart ::= SEQUENCE {
        parameters              TeletexParameters,
        data                    TeletexData }

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 shall be present if (but not only if) the body contains two or more teletex body parts.

The data component is the document, a sequence of teletex strings, each of which encodes one of its pages.

Note 1 – The number-of-pages component identifies the number of elements in the sequence that constitutes the data components and is thus redundant.

Note 2 – 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.

7.3.6 *Videotex*

A **videotex** body part represents videotex data. It has parameters and data components.

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

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

VideotexData ::= VideotexString
```

The parameters component comprises the following parameters:

a) **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.

7.3.7 *Encrypted*

An **encrypted** body part represents the result of encrypting a body part of a type defined by this Recommendation. It has parameters and data components.

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

EncryptedParameters ::= SET -- for further study

EncryptedData ::= BIT STRING -- for further study
```

The parameters of such a body part, and the encryption technique that those parameters might identify and parameterize, are for further study.

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 Recommendation X.209.

7.3.8 *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 }

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**.

Note 1 – The possible future inclusion of a message identifier in the parameters component is for further study. Its present omission provides compatibility with Recommendation X.420 (1984).

Note 2 – That the IPM and purported delivery envelope of a message body part are, in any sense, genuine is unverified.

7.3.9 *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 protocol elements which describe the document's layout structure.

```
MixedModeBodyPart ::= SEQUENCE OF ProtocolElement
```

7.3.10 *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 OctetString.

```
BilaterallyDefinedBodyPart ::= OCTET STRING
```

Note – The use of this body parts is discouraged. It predates the externally defined body part type and is retained for backward compatibility with Recommendation X.420 (1984). The externally defined body part type provides the same capabilities and more, 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.

7.3.11 *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.

```
NationallyDefinedBodyPart ::= ANY
```

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

Note 2 – The use of this body part type is discouraged. It predates the externally defined body part type and is retained for backward compatibility with Recommendation X.420 (1984). The externally defined 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.

7.3.12 *Externally defined*

An **externally defined** body part represents an information object whose semantics and abstract syntax are denoted by an object identifier which the body part carries. It has parameters and data components.

```
ExternalDefinedBodyPart ::= SEQUENCE {
        parameters          [0] ExternallyDefinedParameters OPTIONAL
        data                    ExternallyDefinedData }

ExternallyDefinedParameters ::= EXTERNAL

ExternallyDefinedData ::= EXTERNAL
```

The parameters and data components are externals (see § 32 of Recommendation X.208). Their direct-reference components shall be present, their indirect-reference and data-value-descriptor components absent.

On the basis of the externally defined body part type, all body part types are divided into two important classes as follows:

a) **basic**: Said of any body part type except externally defined. Denoted by an integer (an ASN.1 context-specific tag).

   All basic body part types are defined in this Recommendation.

b) **extended**: Some of the externally defined body part type restricted to any one value of the direct-reference component of the data component of such a body part. Denoted by an object identifier.

   Some (but not necessarily all) extended body part types are defined in Annex B of this Recommendation.

Every extended body part type this Recommendation defines is defined by means of the following macro. Every extended body part type defined elsewhere shall be so defined as well.

```
EXTENDED-BODY-PART-TYPE MACRO ::=
BEGIN
        TYPE   NOTATION ::= Parameters Data
        VALUE  NOTATION ::= value (VALUE OBJECT IDENTIFIER)

        Paramaters          ::= "PARAMETERS" type "IDENTIFIED" "BY" value (OBJECT
                                IDENTIFIER) | empty
        Data                ::= "DATA" type

END
```

An instance of the macro's type notation defines, by means of its PARAMETERS clause, the type of the data value that is represented by the parameters component of such an (externally defined) body part (an external), and the object identifier that appears in the direct-reference component of this parameters component. The omission of the parameters component is implied by the omission of this clause. An instance of the type notation also defines, by means of its DATA clause, the type of the data value that is represented by the data component of such a body part (an external).

An instance of the macro's value notation defines the object identifier that appears as the direct-reference component of the data component of such an (externally defined) body part. The object identifier identifies the encoding rules for the body part. Those body parts whose types this Recommendation defines shall be encoded using ASN.1 basic encoding rules.

Note 1 – This body part enables the exchange of information objects of all kinds, each unambiguously and uniquely identified. This identification relies upon the direct-reference component mentioned above, which is an object identifier. object identifiers are easily obtained, e.g., by national bodies and private organizations.

Note 2 – If an externally defined body part has a parameters component, the object identifier in its direct-reference component is allocated at the same time and by the same naming authority as that in the direct-reference component of the data component.

Note 3 – Like body parts of other types, an externally defined body part may be subjected to conversion. However, specification of the conversion algorithms may be outside the scope of Recommendation X.408.

Note 4 – The basic body part exists for purely historical reasons, predating the externally defined body part type.

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}}
```

An IPN may take either 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 non-receipt-fields chosen*

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 receipt-fields chosen*

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 sequence 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 O/R descriptor whose formal-name component is present and has as its value either an O/R name of the preferred recipient as a result of which the subject IPM was delivered to the user on whose behalf the examination is performed or, if the IPM reached the user because of his membership in a DL, an O/R name appearing in the message's DL expansion history (see § 8.3.1.1.1.7 of Recommendation X.411).

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.

The structure of an IPN is depicted in Figure 2/X.420.

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-preferred-recipient [2] IPMPreferredRecipientField OPTIONAL,
        conversion-eits             ConversionEITsField OPTIONAL}
```

8.1.1 *Subject IPM*

The **subject IPM** common field (M) identifies 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 O/R descriptor.

IPNOriginatorField ::= ORDescriptor

If the IPN's originator is a preferred recipient of the subject IPM, the O/R descriptor above shall be precisely that which is the value of the recipient component of the subject recipient specifier.

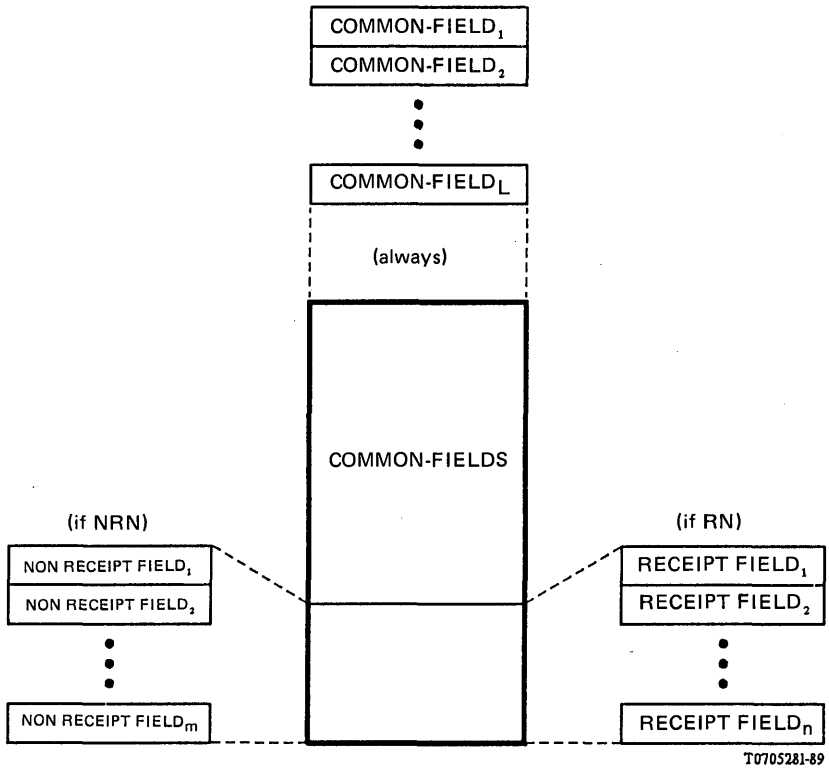


FIGURE 2/X.420

An interpersonal notification

8.1.3 *IPM preferred recipient*

the **IPM preferred recipient** common field (C) identifies the preferred recipient of the subject IPM who gives rise to its delivery to the IPN's originator (an alternate, (DL) member, or substitute recipient). It comprises an O/R descriptor.

IPMPreferredRecipientField ::= ORDescriptor

The O/R 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, it would identify a user other than the IPN's originator or a DL.

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

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

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}
```

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 one of the following values:

a) *ipm-discarded:* The IPM was discarded. This case is further illumined by the *discard reason* field.

b) *ipm-auto-forwarded:* The IPM was auto-forwarded. This case is further illumined 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).

```
DiscardReasonField ::= ENUMERATED {
        ipm-expired                 (0),
        ipm-obsoleted               (1),
        user-subscription-terminated (2) }
```

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-obsoleted: auto-discard* was in effect, obsolete IPMs were being discarded, and the obsoleted 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.

This conditional field shall be present only if the non-receipt reason field has the value *ipm-discarded.*

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 K), 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.

8.3 *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 DEFAULT "" }
```

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

SECTION 3 – ABSTRACT SERVICE DEFINITION

9 Overview

This section defines the abstract service that characterizes interpersonal messaging, and describes the environment in which service is supplied and consumed. It does both using the abstract service definition conventions of Recommendation X.407.

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).**

```
ipme OBJECT
        ::= id-ot-ipme
```

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

```
ipme-refinement REFINE ipme AS
        ipms
            origination   [S] PAIRED WITH ipms-user
            reception     [S] PAIRED WITH ipms-user
            management    [S] PAIRED WITH ipms-user
        ipms-user-RECURRING
        ::= id-ref-primary
```

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/X.420.

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

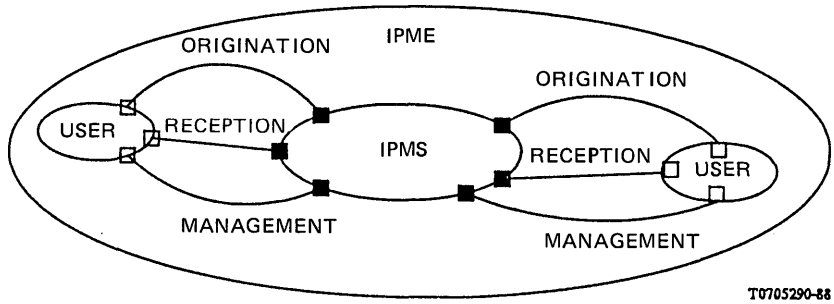


FIGURE 3/X.420

The interpersonal messaging environment

10.1 *Interpersonal messaging system user*

An **interpersonal messaging system user (IPMS user)** is a user that engaged in interpersonal messaging. An IPMS user originates, receives, or both originates and receives information objects of the types defined in § 2.

```
ipms-user OBJECT
        PORTS {
            origination   [C],
            reception     [C],
            management    [C] }
        ::= id-ot-ipms-user
```

The IPME comprises any number of of IPMS users.

Note 1 – As its name suggests, interpersonal messaging is typically an activity of people. Often, therefore, this Recommendation 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.

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

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 OBJECT
        PORTS {
            origination   [S],
            reception     [S],
            management    [S] }
        ::= id-ot-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 § 16 to follow, the IPMS is decomposed into still lesser objects, among which is the MTS. This fact is anticipated in the present paragraph 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 two. 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 two. 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 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 § 13.

Note 1 – The IPMS abstract service involves neither abstract bind nor abstract unbind operations.

Note 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, for example, that the user is an IPMS subscriber. Authentication, where required, is implicit (rather than explicit) in the definition of the IPMS abstract service.

Note 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 two. A user interface need not provide commands in one-to-one correspondence to the service's abstract operations, nor indeed even divide the labor between the user and the IPMS as the service does.

Note 4 – In § 16 to follow, the IPMS is decomposed into objects among which is the MTS. The present paragraph 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{
            OriginateProbe,
            OriginateIPM,
            OriginateRN }
        ::= 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.

```
OriginateProbe ::= ABSTRACT-OPERATION
        ARGUMENT SET{
            envelope  [0] ProbeSubmissionEnvelope,
            content   [1] IPM }
        RESULT SET{
            submission-identifier     [0] ProbeSubmissionIdentifier,
            submission-time           [1] ProbeSubmissionTime }
        ERRORS{
            SubscriptionError,
            RecipientImproperlySpecified }
```

This abstract operation has the following arguments:

a) **envelope** (M): A probe submission envelope, whose makeup 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 O/R names of the preferred 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:

1) **submission-identifier** (M): The probe submission identifier the MTS assigns to probe.

2) **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.

```
OriginateIPM ::= ABSTRACT-OPERATION
        ARGUMENT SET{
            envelope  [0] MessageSubmissionEnvelope,
            content   [1] IPM }
        RESULT SET{
            submission-identifier     [0] MessageSubmissionIdentifier,
            submission-time           [1] MessageSubmissionTime }
        ERRORS{
            SubscriptionError,
            RecipientImproperlySpecified }
```

This abstract operation has the following arguments:

a) **envelope** (M): A message submission envelope, whose makeup 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 O/R names of the preferred 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:

1) **submission-identifier** (M): The message submission identifier the MTS assigns to the submission.

2) **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 RN.

```
OriginateRN ::= ABSTRACT-OPERATION
        ARGUMENT SET{
            envelope   [0] MessageSubmissionEnvelope,
            content    [1] RN}
        RESULT SET{
            submission-identifier      [0] MessageSubmissionIdentifier,
            submission-time            [1] MessageSubmissionTime}
        ERRORS{
            SubscriptionError,
            RecipientImproperlySpecified}
```

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.

This abstract operation has the following arguments:

a) **envelope** (M): A message submission envelope, whose makeup 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). Implicit conversion shall be prohibited, priority that of the subject IPM.

   ii) The O/R names of the preferred recipients and the per-recipient options (i.e., explicit conversion and extensions) desired for each. Reports shall not be requested.

b) **content** (M): The RN being originated.

This abstract operation has the following results:

1) **submission-identifier** (M): The message submission identifier the MTS assigns to the submission.

2) **submission-time** (M): The date and time the message was directly submitted.

12.2 *Reception abstract operatons*

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

```
reception PORT
        SUPPLIER INVOKES{
            ReceiveReport,
            ReceiveIPM,
            ReceiveRN,
            ReceiveNRN}
        ::= id-pt-reception
```

Note 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.

Note 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.

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

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

a) A probe concerning a message whose content was an IPM that was originated with the originate probe abstract operation.

b) A message whose content was an NRN originated as a result of auto-discard or auto-forward.

c) A message whose content was an RN that was originated with the originate RN abstract operations by *auto-acknowledgment.*

d) A message whose content was an IMP that was originated with the originate IPM abstract operation or by *auto-forwarding.*

This abstract operation has the following arguments:

1) **envelope** (M): A report delivery envelope, whose makeup the MTS abstract service defines.

2) **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 agreement 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 requestd. 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.

```
ReceiveIPM ::= ABSTRACT-OPERATION
        ARGUMENT SET{
            envelope  [0] MessageDeliveryEnvelope,
            content   [1] IPM}
        RESULT
        ERRORS {}
```

This abstract operation has the following arguments:

a) **envelope** (M): The message's delivery envelope.

b) **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.

```
ReceiveRN ::= ABSTRACT-OPERATION
        ARGUMENT SET{
            envelope  [0] MessageDeliveryEnvelope,
            content   [1] RN}
        RESULT
        ERRORS {}
```

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.

```
ReceiveNRN ::= ABSTRACT-OPERATION
        ARGUMENT SET {
            envelope  [0] MessageDeliveryEnvelope,
            content   [1] NRN }
        RESULT
        ERRORS { }
```

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.

12.3 *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 {
            ChangeAutoDiscard,
            ChangeAutoAcknowledgment,
            ChangeAutoForwarding }
        ::= 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.

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

When it auto-discards an IPM, the IPMS originates an NRN on the user's behalf if, and only if, one was requested of him by means of the notification-requests component of the subject recipient specifier.

This abstract operation has the following arguments:

a) **auto-discard-expired-IPMs** (M): Whether or not expired IPMs are to be auto-discarded. A Boolean.

b) **auto-discard-obsolete-IPMs** (M): Whether or not obsolete IPMs are to be auto-discarded. A Boolean.

This abstract operation has no results.

12.3.2 *Change auto-acknowledgment*

The **change auto-acknowledgment** abstract operation enables or disables **auto-acknowledgment**, the automatic origination of RNs by the IPMS on the user's behalf. Such origination occurs upon delivery of IPMs that request RNs of the user by means of the notification-requests components of their subject recipient specifiers.

```
ChangeAutoAcknowledgment ::= ABSTRACT-OPERATION
        ARGUMENT SET {
            auto-acknowledge-IPMs                [0] BOOLEAN,
            auto-acknowledge-suppl-receipt-info  [1]
            SupplementaryInformation OPTIONAL }
RESULT
ERRORS {
        SubscriptionError }
```

This abstract operation has the following arguments:

a) **auto-acknowledge-IPMs** (M): Whether or not IPMs are to be auto-acknowledged. A Boolean.

b) **auto-acknowledge-suppl-receipt-info** (C): The suppl receipt info receipt field of each RN provoked by auto-acknowledgment.

   This conditional argument shall be present if, and only if, the auto-acknowledge-IPMs argument has the value *true*.

This abstract operation has no results.

12.3.3 *Change auto-forwarding*

The **change auto-forwarding** abstract operation enables or disables **auto-forwarding**, the automatic forwarding of IPMs by the IPMS to pre-specified users or DLs. Such forwarding occurs upon delivery of the IPMs.

```
ChangeAutoForwarding ::= ABSTRACT-OPERATION
        ARGUMENT SET {
            auto-forward-IPMs          [0] BOOLEAN,
            auto-forward-recipients    [1] SEQUENCE OF ORName OPTIONAL,
            auto-forward-heading       [2] Heading OPTIONAL,
            auto-forward-comment       [3] AutoForwardComment OPTIONAL }
RESULT
ERRORS {
        SubscriptionError,
        RecipientImproperlySpecified }
```

The body of each IPM the IPMS originates as a result of auto-forwarding comprises a single body part of type message. The content of the message represented by that body part is the forwarded IPM.

When it auto-forwards an IPMS, the IPMS originates an NRN on the user's behalf if, and only if, one was requested of him by means of the notification-requests component of the subject recipient specifier.

This abstract operation has the following arguments:

a) **auto-forward-IPMs** (M): Whether or not IPMs are to be auto-forwarded. A Boolean.

b) **auto-forward-recipients** (C): The users or DLs to which IPMs are to be auto-forwarded. A sequence of O/R names.

   This conditional argument shall be present if, and only if, the auto-forward-IPMs argument has the value *true*.

c) **auto-forward-heading** (C): The heading that is to be used for each forwarding IPM. Its auto-forwarded heading field shall have the value *true*.

   This conditional argument shall be present if, and only if, the auto-forward-IPMs argument has the value *true*.

d) **auto-forward-comment** (C): The value that is to be supplied as the auto-forward comment non-receipt field of each NRN conveyed to the originator of an auto-forwarded IPM.

   The conditional argument shall be present if, and only if, the auto-forward-IPMs argument has the value *true*.

This abstract operation has no results.

Note – This abstract operation is intended to define the essence of auto-forwarding, sophisticated auto-forwarding capabilities, e.g., like those of an MS.

13 Abstract errors

The abstract errors that may be reported in response to the invocation of the abstract operations available at origination, reception, and management ports are defined and described below or as part of the MTS abstract service definition.

Note – The set of abstract errors represented below is intended to be illustrative, rather than exhaustive.

13.1 *Subscription error*

The **subscription error** abstract error reports that the user has not subscribed to one or more of the elements of service implicit in his invocation of the abstract operation whose performance is aborted.

```
SubscriptionError ::= ABSTRACT-ERROR
        PARAMETER SET {
            problem  [0] SubscriptionProblem }
```

This abstract error has the following parameters:

a) **problem** (M): The subscription-related problem encountered.

```
SubscriptionProblem ::= ENUMERATED {
    ipms-eos-not-subscribed(0),
    mts-eos-not-subscribed (1) }
```

This parameter may assume any one of the following values:

i) *IPMS-eos-not-subscribed*: An IPMS element of service is not subscribed.

ii) *MTS-eos-not-subscribed*: An MTS element of service is not subscribed.

13.2 *Recipient improperly specified*

The **recipient improperly specified** abstract error reports that one or more of the O/R names supplied as arguments of the abstract operation whose performance is aborted, or as components of its arguments, are invalid.

This abstract error is defined by the MTS abstract service.

14 Other capabilities

In addition to the capabilities embodied in the IMPS abstract service, defined above, the IPMS shall transparently extend to each use the other MS and MTS capabilities identified below. (The enumeration of these capabilities necessarily anticipates the fact, stated in § 16, that MSs and the MTS are among the IPMS' component parts.)

The following additional capabilities shall be provided:

a) *Submission:* Capabilities of the MS' or MTS' submission port not embodied in the IPMS abstract service e.g., the ability to cancel delivery of a previously originated message whose content is an IPM (but not an RN), if deferred delivery was selected.

b) *Delivery:* Capabilities of the MTS' delivery port not embodied in the IPMS abstract service, e.g., the ability to temporarily control the kinds of information objects the MTS conveys to the user's *UA*.

c) *Administration:* The capabilities of the MS's or MTS's administration port.

d) *Retrieval:* The capabilities of the MS' retrieval port.

In addition to the above and as a local matter, the IPMS may provide to users additional capabilities neither defined nor limited by this Recommendation. Among such capabilities are those of the directory.

Note – The required capabilities of this clause are excluded from the formal definition of the IPMS abstract service for purely pragmatic reasons, in particular, because their inclusion would largely and needlessly reproduce the definitions of the MS and MTS abstract operations upon which the capabilities are based.

SECTION 4 – ABSTRACT SERVICE PROVISION

15 Overview

This section specifies how the IPMS provides the IPMS abstract service to users.

This section covers the following topics:

a) secondary object types,

b) secondary port types,

c) user agent operation,

d) message store operation,

e) message contents,

f) port realization,

g) conformance.

16 Secondary object types

The IPMS can be modeled as comprising lesser objects which interact with one another by means of (additional) ports.

```
ipms-refinement REFINE ipms AS
        mTS
            submission      [S]  PAIRED WITH ipms-ua, ipms-ms
            delivery        [S]  PAIRED WITH ipms-ua, ipms-ms
            administration  [S]  PAIRED WITH ipms-ua, ipms-ms
        ipms-ua RECURRING
            origination     [S]  VISIBLE
            reception       [S]  VISIBLE
            management      [S]  VISIBLE
        ipms-ms RECURRING
            submission      [S]  PAIRED WITH ipms-ua
            retrieval       [S]  PAIRED WITH ipms-ua
            administration  [S]  PAIRED WITH ipms-ua
        tlma RECURRING
            origination     [S]  VISIBLE
            reception       [S]  VISIBLE
            management      [S]  VISIBLE
        tlxau RECURRING
            origination     [S]  VISIBLE
            reception       [S]  VISIBLE
            management      [S]  VISIBLE
        pdau RECURRING
            reception       [S]  VISIBLE
        ::= id-ref-secondary
```

These lesser objects are referred to as the **secondary objects** of interpersonal messaging. They include a single, central object, the MTS, and numerous peripheral objects: *interpersonal messaging system user agents (IPMS UAs), interpersonal messaging system message stores (IPMS MSs), telematic agents (TLMAs), telex access units (TLXAUs)*, and PDAUs.

The structure of the IPMS is depicted in Figure 4/X.420. As shown by Figure 4/X.420, *IPMS UAs, TLMAs, TLXAUs*, and PDAUs are the instruments by means of which the IPMS provides the IPMS abstract service to users.

The secondary object types are defined and described below. The types of ports by means of which they interact are discussed in § 17.

Note 1 – The refinement above encompasses all possible interconnection of all possible objects. It ignores the possible absence of objects of a particular type (e.g., PDAU), and specific logical configurations of the *IPMS MS*. The latter are identified in Recommendation X.402.

Note 2 – Recommendation T.330 effectively extends the abstract service of Interpersonal Messaging by its definition of a *miscellanea* port, which is not shown in the figure. See the note in § 16.3.

Note 3 – The MTS supplies import and export ports. However, since those ports are not formally defined (in Recommedation X.411), they are not included in the formal refinement above.

16.1 *Interpersonal messaging system user agent*

An **interpersonal messaging system user agent (IPMS UA)** is a UA tailored so as to better assist a single user to engage in interpersonal messaging. It helps him originate, receive, or both originate and receive messages containing information objects of the types defined in section two.

```
ipms-us OBJECT
        PORTS {
            origination     [S],
            reception       [S],
            management      [S],
            submission      [C],
            delivery        [C],
            retrieval       [C],
            administration  [C]}
        ::= id-ot-ipms-ua
```

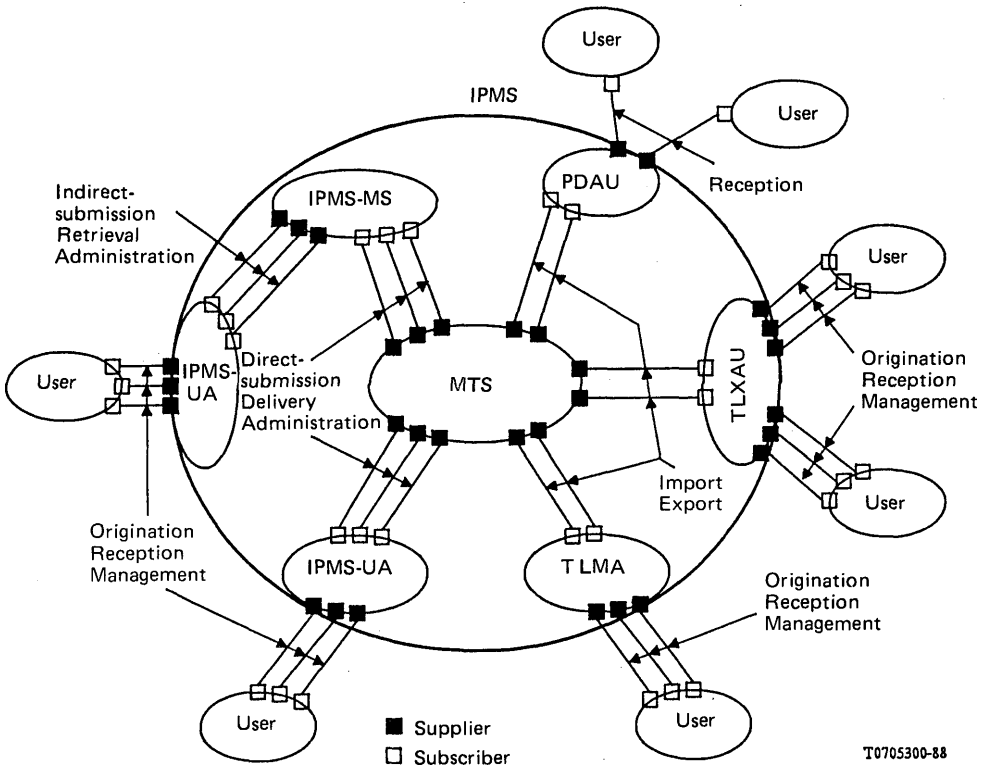


FIGURE 4/X.420

The interpersonal messaging system

The IPMS comprises any number of IPMS UAs.

Note – For brevity, the term "UA" is used throughout the rest of this Recommendation with the meaning of "IPMS UA".

16.2 *Interpersonal messaging system message store*

An **interpersonal messaging system message store (IPMS MS)** is an MS tailored so as to better assist a single UA engage in interpersonal messaging. It helps it submit, take delivery of, or both submit and take delivery of messages containing information objects of the types defined in Section 2.

```
ipms-ms OBJECT
        PORTS {
            submission      [S],
            retrieval       [S],
            administration  [S],
            submission      [C],
            delivery        [C],
            administration  [C] }
        ::= id-ot-ipms-ms
```

The IPMS comprises any number of IPMS MSs.

Note – For brevity, the term "MS" is used throughout the rest of this Recommendation with the meaning of "IPMS MS".

16.3 *Telematic agent*

A **telematic agent (TLMA)** is an AU that helps a single indirect user engage in interpersonal messaging from a telematic terminal, along with that terminal and the network that joins the two. A TLMA helps the user originate, receive, or both originate and receive messages containing information objects of the types defined in Section 2.

```
tlma OBJECT
        PORTS {
            origination     [S],
            reception       [S],
            management      [S],
            miscellanea     [S] }
        ::= id-ot-tlma
```

The IPMS comprises any number of TLMAs.

Note 1 – A TLMA consumes import and export ports. However, since those ports are not formally defined (in Recommendation X.411), they are not included in the formal definition of TLMA above.

Note 2 – A TLMA's *miscellanea* port is defined in Recommendation T.330. It is not part of the IPMS abstract service in its most general form, which is the subject of this Recommendation. Rather it embodies capabilities available only to a TLMA user. For this reason, it is not considered further here and is not included in the formal refinement of the IPMS found in § 16.

16.4 *Telex access unit*

A **telex access unit (TLXAU)** is an AU that helps any number of indirect users engage in interpersonal messaging from telex terminals. It helps them originate, receive, or both originate and receive messages containing information objects of the types defined in section two.

```
tlxau OBJECT
        PORTS {
            origination     [S],
            reception       [S],
            management      [S] }
        ::= id-ot-tlxau
```

The IPMS comprises any number of TLXAUs.

Note – A TLXAU consumes import and export defined (in Recommendation X.411) they are not included in the formal definition of TLXAU above.

16.5 *Physical delivery access unit*

In the present context, a PDAU helps any number of indirect users engage in interpersonal messaging by means of a PDS. It helps them receive (but not originate) messages containing information objects of the types defined in section two.

```
pdau OBJECT
        PORTS {
            reception       [S] }
        ::= id-ot-pdau
```

The IPMS comprises any number of PDAUs.

Note – A PDAU consumes import and export ports. However, since those ports are not formally defined (in Recommendation X.411), they are not included in the formal definition of PDAU above.

16.6 *Message transfer system*

In the present context, the MTS conveys information objects of the types defined in Section 2 between UAs, MSs, TLMAs, and AUs.

The IPMS comprises a single MTS.

17 Secondary port types

The secondary objects of interpersonal messaging are joined to and interact with one another by means of ports. These ports, which MSs and the MTS supply, are referred to as the **secondary ports** of interpersonal messaging. They are of the types identified below.

The capabilities embodied in one submission, one retrieval, and one administration port constitute the MS abstract service. They are defined in Recommendation X.413.

The capabilities embodied in one submission, one delivery, and one administration port constitute the MTS abstract service. They are defined in Recommendation X.411.

Note – By means of the abstract bind operation which guards its ports, an MS or the MTS typically authenticates another secondary object before offering its abstract service to that object.

17.1 *Submission*

In the present context, a submission port is the mean by which a UA (directly or indirectly) or an MS (directly) submits probes concerning and messages containing information objects of the types defined in section two.

An MS supplies one submission port to its UA.

The MTS supplies one submission port to each UA configured without an MS and to each MS.

17.2 *Delivery*

In the present context, a delivery port is the means by which a UA or MS takes delivery of reports concerning and messages containing information objects of the types defined in Section 2.

The MTS supplies one delivery port each UA configured without an MS and to each MS.

17.3 *Retrieval*

In the present context, a retrieval port is the means by which a UA retrieves reports concerning and messages containing information objects of the types defined in Section 2.

An MS supplies one retrieval port to its UA.

17.4 *Administration*

In the present context, an administration port is the means by which a UA changes information about itself or its user on file with its MS, or a UA or MS changes such information on file with the MTS.

An MS supplies one administration port to its UA.

The MTS supplies one administration port to each UA configured without an MS and to each MS.

17.5 *Import*

In the present context, an import port is the means by which the MTS imports reports concerning and messages containing information objects of the types defined in Section 2.

The MTS supplies one import port to each AU (or TLMA).

17.6 *Export*

In the present context, an export port is the means by which the MTS exports probes concerning and messages containing information objects of the types defined in Section 2.

The MTS supplies one export port to each AU (or TLMA).

18 User agent operation

A UA must employ the MTS in a particular way in order to (correctly) provide the IPMS abstract service to its user. If the user is equipped with an MS, the latter contributes to the provision of the abstract service and, therefore, is subject to the same rules.

The rules that govern the operation of a UA (and MS) are the subject of the present clause. The operation of a TLMA or AU is beyond the scope of this Recommendation.

Note 1 – It is for historical reasons that the Recommendation that defines the IPMS abstract service also specifies how a UA (and MS), but not a TLMA or AU, provides it.

Note 2 – The purpose of this clause is not to dictate or constrain the implementation of a real UA unnecessarily, but rather to clarify the meaning and intended effect of the IPMS abstract service.

18.1 *State variables*

The operation of a UA is described below with the aid of *state variables.* A **state variable** is an information item whose value records the results of the UA's past interactions with its user and influences future interactions. State variables are common to (i.e., shared by) the UA's origination, reception, and management ports.

The UA maintains each state variable continuously, i.e., throughout its user's IPMS subscription. Each Boolean state variable is assigned the value *false* when the subscription commences. The initial values of other state variables are immaterial and therefore unspecified.

The UA alters its state variables when performing or invoking abstract operations. It consults them in determining how to perform, or whether or how to invoke abstract operations. Their values (if any) transcend the binding and unbinding of ports.

Note – State variables are pedagogic devices not intended to constrain the implementation of a real UA unnecessarily. In particular, a UA need not maintin run-time data structures corresponding to the state variables if the behaviour required of it can be assured in another way.

18.2 *Performance of origination operations*

A UA shall perform the abstract operations it makes available at its origination port as prescribed below. The UA alters none of its state variables in the performance of these particular operations.

In the performance of these operations, the UA invokes the following abstract operations of the MTS abstract service (which, for the remainder of this paragraph, are unqualified as to their source):

a) probe submission,

b) message submission.

Note – In response to the invocation of these abstract operations, a UA reports abstract errors as appropriate. Specification of the precise circumstances under which each abstract error should be reported is beyon the scope of this Recommendation.

18.2.1 *Originate probe*

A UA shall perform the originate probe abstract operation by invoking probe submission with the arguments indicated below, and by returning to its user the results indicated below.

The arguments of probe submission shall be as follows:

a) *Envelope*: The components of this argument that constitute per-probe fields shall be as follows;

   i) *Originator-name:* The O/R name of the UA's user.

   ii) *Content-type, content-length,* and *original-encoded-information-types:* Determined from originate probe's content argument as specified in §§ 20.2 to 20.4.

   iii) *Content-identifier:* Specified or omitted as a local matter.

   The components of this argument that constitute per-recipient fields shall be as specified by originate probe's envelope argument.

The result of originate probe shall be as follows:

1) *Submission-identifier:* Probe submission's probe-submission-identifier result.

2) *Submission-time:* Probe submission's probe-submission-time result.

Note 1 – The UA shall ignore all properties of originate probe's content argument other than those mentioned above.

Note 2 – How the UA employs probe submission's content-identifier result is a local matter.

18.2.2 *Originate IPM*

A UA shall perform the originate IPM abstract operation by invoking message submission with the arguments indicated below, and by returning to its user the results indicated below.

The arguments of message submission shall be as follows:

a) *Envelope:* The components of this argument that constitute per-message fields shall be as follows; those not explicitly mentioned below shall be as specified by originate IPM's envelope argument:

   i) *Originator-name:* The O/R name of the UA's user.

   ii) *Content-type* and *original-encoded-information-types:* Determined from originate IPM's content argument as specified in §§ 20.4 and 20.4, respectively.

   iii) *Content-identifier:* Specified or omitted as a local matter.

   The components of this argument that constitute per-recipient fields shall be as specified by originate IPM's envelope argument.

b) *Content:* Determined from originate IPM's content argument (identified as an IPM) as specified in § 20.1.

If the blind copy recipients heading field of the IPM identifies one or more users and DLS, the UA shall invoke message submission multiple times, upon each occasion varying the heading field so as to comply with the information hiding requirements of § 7.2.6.

The results of originate IPMS shall be as follows:

1) *Submission-identifier:* Message submission's message-submission-identifier result.

2) *Submission-time:* Message submission's message-submission-time result.

Note 1 – How the UA employs message submission's content-identifier result is a local matter.

Note 2 – The inclusion of message submission's extensions result among originate IPM's results in proper and for further study.

18.2.3 *Originate RN*

A UA shall perform the originate RN abstract operation by invoking message submission with the arguments indicated below, and by returning to its user the results indicated below.

The arguments of message submission shall be as follows:

a) *Envelope:* The components of this argument that constitute per-message fields shall be as follows; those not explicitly mentioned below shall be as specified by originate RN's envelope argument:

   i) *Originator-name:* The O/R name of the UA's user.

   ii) *Content-type* and *original-encoded-information-types:* Determined from the RN as specified in §§ 20.2 and 20.4, respectively.

   iii) *Content-identifier:* Specified or omitted as a local matter.

   iv) *Deferred-delivery-time:* Omitted.

   The components of this argument that constitute per-recipient fields shall be as specified by originate RN's envelope argument.

b) *Content:* Determined from originate RN's content argument (identified as an RN) as specified in § 20.1.

The result of originate RN shall be as follows:

1) *Submission-identifier:* Message submission's message-submission-identifier result.

2) *Submission-time:* Message submission's message-submission-time result.

Note 1 – How the UA employs message submission's content-identifier result is a local matter.

Note 2 – The inclusion of message submission's extensions result among originate RN's results is proper and for further study.

18.3 *Performance of management operations*

A UA shall perform the abstract operations it makes available at its management port as specified below. The UA alters one or more of its state variables (see below) in the performance of each operation.

Note – In response to the invocation of these abstract operations, a UA reports abstract errors as appropriate. Specification of the precise circumstances under which each abstract error should be repeated is beyond the scope of this Recommendation.

18.3.1 *Change auto-discard*

To assist it in providing this abstract operation, a UA maintains the following state variables:

a) **Auto-discard-expired-IPMs**: A Boolean that indicates whether or not auto-discard is in effect for expired IPMs.

b) **Auto-discard-obsolete-IPMs**: A Boolean that indicates whether or not auto-discard is in effect for obsolete IPMs.

A UA shall perform the change auto-discard abstract operation by recording the values of the auto-discard-expired-IPMs and auto-discard-obsolete-IPMs arguments in the correspondingly named state variables.

18.3.2 *Change auto-acknowledgment*

To assist it in providing this abstract operation, a UA maintains the following state variables:

a) **auto-acknowledge-IPMs**: A Boolean that indicates whether or not auto-acknowledgment is in effect.

b) **auto-acknowledge-suppl-receipt-info**: The suppl receipt info field of each RN provoked by auto-acknowledgment.

A UA shall perform the change auto-acknowledgment abstract operation by recording the value of the auto-acknowledge-IPMs argument in the correspondingly named state variable. If the value is *true*, it also shall record the value of the Auto-acknowledge-suppl-receipt-info argument in the correspondingly named state variable.

18.3.3 *Change auto-forwarding*

To assist it in providing this abstract operation, a UA maintains the following state variables:

a) **auto-forward-IPMs**: A Boolean that indicates whether or not auto-forwarding is in effect.

b) **auto-forward-recipients**: A sequence of O/R names that identify the users and DLs to which IPMs are being auto-forwarded.

c) **auto-forward-heading**: The heading of each forwarding IPM provoked by auto-forwarding. Its auto-forwarded field has the value *true*.

d) **auto-forward-comment**: The auto-forward comment non-receipt field of each NRN conveyed to the originator of an auto-forwarded IPM.

A UA shall perform the change auto-forwarding abstract operation by recording the value of the auto-forward-IPMs argument in the correspondingly named state variable. If the value is *true*, it also shall record the values of the auto-forward-recipients, auto-forward-heading, and auto-forward-comment arguments in the correspondingly named state variables.

18.4 *Invocation of reception operations*

A UA shall invoke that abstract operations available at its reception port as specified below. The UA alters none of its state variables in connection with its invocation of these operations.

The UA invokes these operations in response to the MTS' invocation of the following abstract operations of the MTS abstract service (which, for the remainder of this paragraph, are unqualified as to their source):

a) report delivery,

b) message delivery.

Note – The abstract operation of a reception port report no errors.

18.4.1 *Receive report*

Whenever the MTS invokes report delivery at a UA's delivery port, the UA shall invoke the receive report abstract operation with the following arguments:

a) *Envelope:* Report delivery's envelope argument.

b) *Undelivered-object:* Determined from report delivery's returned-content argument as specified in § 20.1.

Note – How the UA employs the content-identifier component of report delivery's envelope argument is a local matter.

18.4.2 *Receive IPM*

Whenever the MTS invokes message delivery at a UA's delivery port, and its content argument encodes an IPM as specified in § 20.1, the UA shall invoke the receive IPM abstract operation with the following arguments, provided that the message is subject to neither auto-forwarding nor auto-discard (see § 18.5):

a) *Envelope:* Message delivery's envelope argument.

b) *Content:* Determined from message delivery's content argument as specified in § 20.1 (but no longer marked as an IPM).

18.4.3 *Receive RN*

Whenever the MTS invokes message delivery at a UA's delivery port, and its content argument encodes in RN as specified in § 20.1, the UA shall invoke the receive RN abstract operation with the following arguments:

a) *Envelope:* Message delivery's envelope argument.

b) *Content:* Determined from message delivery's content argument as specified in § 20.1 (but no longer marked as an RN).

18.4.4 *Receive NRN*

Whenever the MTS invokes message delivery at a UA's delivery port, and its content argument encodes an NRN as specified in § 20.1, the UA shall invoke the receive NRN abstract operation with the following arguments:

a) *Envelope:* Message delivery's envelope argument.

b) *Content:* Determined from message delivery's content argument as specified in § 20.1 (but no longer marked as an NRN).

18.5 *Internal procedures*

A UA shall perform as specified below the internal procedures of auto-discard, -acknowledgment, and -forwarding in ultimate fulfilment of the abstract operations available at its management port.

The procedures involve the following abstract operations of the MTS abstract service (which, for the remainder of this paragraph, are unqualified as to their source):

a) message submission,

b) message delivery.

As implied by the above, in the course of the procedures, the UA has occasion to invoke message submission. What it does with the results of this abstract operation is a local matter.

The UA shall consider as a candidate for each procedure individually every message for which all of the following conditions hold:

a) The MTS has conveyed the message to the UA by invoking message delivery at the UA's delivery port.

b) The UA has not conveyed the message to the user by invoking receive IPM at the user's reception port.

c) The message contains an IPM (rather than an IPN).

Note – With reference to item b) above, the message might be detained in the UA, e.g., as might be typical, because of the user's unavailability.

18.5.1 *Auto-discard*

The UA shall subject to auto-discard each candidate message with respect to whose content either of the following conditions holds:

a) The auto-discard-expires-IPMs state variable has the value *true* and the date and time denoted by the IPM's expiry time field have past.

b) The auto-discard-obsolete-IPMs state variable has the value *true* and another candidate IPM identifies the present candidate IPM by means of its obsoleted IPMs heading field.

The UA shall auto-discard each such message as follows.

18.5.1.1 *Discard of IPM*

The UA shall discard the IPM, so as to never convey it to the user.

18.5.1.2 *Construction of NRN*

The UA shall construct an NRN if, and only if, one is requested by means of the notification-requests component of the IPM's subject recipient specifier.

The NRN shall have the common fields prescribed for auto-acknowledgment (see § 18.5.2.1).

The NRN shall have the following receipt fields:

a) *Non-receipt reason:* The value *ipm-discarded.*

b) *Discard reason:* The value *imp-expired* or *ipm-obsoleted,* whichever applies. If both apply, either value may be specified.

c) *Auto-forward comment:* Omitted.

d) *Returned IPM:* If the IPM's return is requested by means of the notification-requests component of its subject recipient specifier, and the converted-encoded-information-types component of message delivery's envelope argument is absent, IPM. Omitted otherwise.

18.5.1.3 *Submission of NRN*

The UA shall submit the NRN (if any) above by invoking message submission. Its envelope argument shall be as prescribed for auto-acknowledgment (see § 18.5.2.2), its content argument determined from the NRN as specified in § 20.1.

18.5.2 *Auto-acknowledgment*

The UA shall subject to auto-acknowledgment each candidate message with respect to whose content the following condition holds:

a) The auto-acknowledgment state variable has the value *true* and the IPM requests an RN of the UA's user by means of the notification-requests component of the IPM's subject recipient specifier.

The UA shall auto-acknowledge each such message as follows.

18.5.2.1 *Construction of RN*

The UA shall contruct an RN.

The RN shall have the following common fields:

a) *Subject IPM:* The IPM's This IPM heading field.

b) *IPN originator:* Specified or omitted as a local matter (but, of course, in accordance with § 8.1.2).

c) *IPM preferred recipient:* The recipient component of the IPM's subject recipient specifier, unless its formal-name component is the O/R name of the UA's user, in which case the field shall be omitted.

d) *Conversion EITs:* Converted-encoded-information-types component of message delivery's envelope argument.

The RN shall have the following receipt field:

a) *Receipt time:* The current date and time.

b) *Aknowledgment mode:* The value automatic.

c) *Suppl receipt info:* The auto-acknowledge-suppl-receipt-info state variable.

18.5.2.2 *Submission of RN*

The UA shall submit the RN above by invoking message submission with the following arguments:

a) *Envelope:* The components of this argument shall be as prescribed for performance of the originate RN abstract operation with the following exceptions:

   i) *Priority:* As specified by message delivery's envelope argument.

   ii) *Per-message-indicators:* A local matter, except that *conversion-prohibited* shall be among the values specified.

   iii) *Per-recipient-fields:* A single field whose recipient-name component shall be the originator-name component of message delivery's envelope argument. Reports shall not be requested.

b) *Content:* Determined from the RN as specified in § 20.1.

18.5.3 *Auto-forwarding*

The UA shall subject to auto-forwarding every candidate message, provided that the auto-forward-IPMs state variable has the value *true.*

The UA shall auto-forward each such message as follows.

18.5.3.1 *Prevention of loops*

The UA shall suppress auto-forwarding if, and only if, the IPM to be forwarded itself contains a forwarding IPM that the UA previously created. Auto-forwarding shall be suppressed whether the forwarding IPM appears (directly) in a message bodypart of the IPM to be forwarded, or (nested) in a message body part of the IPM that appears in such a body part.

The UA shall consider itself to have created the forwarding IPM above (whose auto-forwarded heading fields has the value *true*) if, and only if, the originator-name component of the IPM's parameters component matches the O/R name of the UA's user.

Note – Auto-forwarding an IPM of the kind described above would constitute an auto-forwarding "loop".

18.5.3.2 *Construction of IPM*

The UA shall construct a forwarding IPM whose heading is the auto-forward-heading state variable (its auto-forwarded field having the value *true*) and whose body comprises a single body part of type message.

The message body part shall have the following components:

a) *Parameters:* The envelope argument of message delivery.

b) *Data:* The IPM to be forwarded.

18.5.3.3 *Submission of IPM*

The UA shall submit the IPM it constructed above by invoking message submission with the following arguments:

a) *Envelope:* The components of this argument shall be as follows:

   i) *Originator-name:* The O/R name of the UA's user.

   ii) *Content-type* and *original-encoded-information-types:* Determined from the IPM as specified in §§ 20.2 and 20.4.

   iii) *Content-identifier:* Specified or omitted as a local matter.

   iv) *Priority:* As specified by message delivery's envelope argument.

   v) *Per-message-indicators* and *extensions:* A local matter.

   vi) *Deferred-delivery-time:* Omitted.

   vii) *Per-recipient-fields:* Their recipient-name components shall be the O/R names that make up the auto-forward-recipients state variable. Their other components are a local matter.

b) *Content:* Determined from the IPM as specified in § 20.1.

18.5.3.4 *Construction of NRN*

The UA shall construct an NRN if, and only if, one is requested by means of the notification-requests component of the forwarded IPM's subject recipient specifier.

The NRN shall have the common fields prescribed for the performance of auto-acknowledgment.

The NRN shall have the following receipt fields:

a) *Non-receipt reason:* The value *ipm-auto-forwarded.*

b) *Discard reason:* Omitted.

c) *Auto-forward comment:* The auto-forward-comment state variable.

d) *Returned IPM:* If the IPM's return is requested by means of the notification-requests component of its subject recipient specifier, and the converted-encoded-information types component of message delivery's envelope argument is absent, the IPM. Omitted otherwise.

18.5.3.5 *Submission of NRN*

The UA shall submit the NRN (if any) above by invoking message submission. Message submission's envelope argument shall be as prescribed for auto-acknowledgment, its content argument determined from the NRN as specified in § 20.1.

19 Message store operation

An MS must perform certain interpersonal messaging-specific functions to qualify as an IPMS MS and thus distinguish itself from a generic MS. These functions are the subject of the present paragraph.

19.1 *Creation of information objects*

An IPMS shall satisfy the following requirements related to the information objects it maintains:

a) The MS shall maintain a separate information object for each (message containing an) IPM or IPN that is delivered to it.

b) The MS shall maintain as a separate information object not only each (message containing a) forwarding IPM (pursuant to Item a)) but also each (message containing a) forwarded IPM (recursively).

c) The MS shall assign sequence numbers depth-first to the messages in the hierarchy formed by a forwarding IPM and its forwarded IPMs.

Example – If IPM *A* contains IPMs *B* and *C* among its body parts, and if IPM *B* contains IPMS *D* and *E* among its body parts, sequence numbers will be assigned to the IPMs in the following order: *A*, *B*, *D*, *E*, and *C*.

19.2 *Maintenance of attributes*

An IPMS MS shall satisfy the following requirements related to MS attributes:

a) For each IPM or IPN it holds, the MS shall support the attributes of Annex C as specified therein.

b) For each IPM it holds, the MS shall give the following messages to the defined values of the MS-status attribute:

   i) *new:* No attribute values have been conveyed to the UA.

   ii) *listed:* At least one attribute value has been conveyed to the UA, and at least one body part has not been conveyed.

   iii) *processed:* All body parts have been conveyed to the UA.

c) For each IPN it holds, the MS shall give the following meanings to the defined values of the MS-status attribute:

   i) *new:* No attribute values have been conveyed to the UA.

   ii) *listed:* At least one attribute value has been conveyed to the UA, and at least one attribute other than Returned IPM has not been conveyed.

   iii) *processed:* All attributes, with the possible exception of returned IPM, have been conveyed to the UA.

d) The MS-status attribute shall reflect the state of affairs prior to an abstract operation invocation that alters its value.

e) The content-type attribute of each (message containing an) IPM or IPN that is delivered to the MS shall have the value id-mct-p2-1984 or id-mct-p2-1988 (see Annex D), as appropriate, depending upon the content type of the delivered message (see § 20.2).

19.3 *Notification of non-receipt*

When it discards an IPM while performing the delete abstract operation of the MS abstract service, the MS shall submit a NRN if one is requested and the IPM's MS-status attribute has the value *listed.*

19.4 *Auto-forwarding*

An IPMS MS shall perform the auto-forward action of Recommendation X.413 as specified in § 18.5.3. It makes use of the other-parameters component of the auto-forward-registration argument of the register MS abstract operation of the MS abstract service. The data type of the other-parameters component is defined as follows:

```
Forwarded Info ::= SET{
        auto-forwarding-comment [0] AutoForwardComment OPTIONAL,
        cover-note [1] IA5TextBodyPart OPTIONAL,
        this-ipm-prefix [2] PrintableString (SIZE
        (1. .ub-ipm-identifier-suffix)) OPTIONAL}
```

In addition, the MS shall satisfy the following requirements:

a) Submit an NRN even if it retains a copy of the forwarded IPM.

b) Draw the NRN's auto-forward comment field, if any, from the other-parameters component.

c) Draw the cover-note, if any, to be included with the forwarded IPM, from the other-parameters component.

d) Prefix the user-relative-identifier component of this IPM field of the forwarding IPM's heading with, if present, this-ipm-prefix.

Note – An (IPMS) MS performs neither auto-discard nor auto-acknowledgment, except possibly as a local matter.

19.5 *Manual forwarding*

An IPMS shall support the manual forwarding of a message using the forwarding-request extension of Recommedation X.413, as specified in § 6.6. The IPMS MS user may submit an IPM, including heading and body, using the message submission operation, and identify by using the forwarding-request extension a message that is already in the MS, and to be combined with the submitted message body for forwarding to the message's recipient.

The submitted message body and the forwarded message are then combined by inserting the forwarded message as a message body part into the submitted message body.

20 Message contents

As has already been seen, various secondary objects (e.g., UAs) have occasion to convey the information objects of section two as the contents of messages, as well as to convey probes concerning such messages. This paragraph specifies precisely how they shall do this.

The rules governing the transmittal of such messages and probes, and the semantics and abstract and transfer syntaxes of their contents, are called the **interpersonal messaging protocol (P2)**.

Note – The name "P2" reflects the historical fact that this was the second message handling protocol to be developed.

20.1 *Content*

A secondary object that submits a message containing an IPM or IPN shall supply as the octets of the octet string that constitutes the content of the message the result of encoding the InformationObject of section two in accordance with the basic encoding rules of Recommendation X.209.

20.2 *Content type*

A secondary object that submits a message containing an IPM or IPN shall select its content type as follows.

If the IPM or IPN satisfies all of the following constraints, the Integer 2 shall be specified:

i) The heading (of an IPM) lacks the extensions field.

ii) The body (of an IPM) lacks externally defined body parts.

iii) The parameters element of any videotex body part: ... of (an IPM) lacks the syntax member.

iv) Every component of the IPM or IPN that is a value of a data type defined as part of the MTS abstract service meets the constraints of Recommendation X.411 (1984).

The types in question are those listed in the IMPORTS clause of the ASN.1 module defined in Annex E. The constraints in question are detailed in an annex of Recommendation X.419.

v) The data element of any message body part (of an IPM) satisfies these same constraints (recursively).

Otherwise, the Integer 22 shall be specified.

Note 1 – The message content protocol (here) denoted by the Integer 2 is identical to that specified by Recommendation X.420 (1984) (as clarified by Version 6 of the *X.400-series Implementor's Guide*), except that the simple formattable document body part type, defined in the latter, is omitted from the former.

Note 2 – The Integer 2 is favoured, above, over the Integer 22 to foster interworking between systems conforming to this Recommendation and systems conforming (only) to Recommendation X.420 (1984).

Note 3 – The MTS does not convert between message content protocols. Thus it does not convert between P2 as defined by this Recommendation alone (and denoted by the Integer 22) and P2 as defined by both this Recommendation and Recommendation X.420 (1984) (and denoted by the Integer 2).

20.3 *Content length*

A secondary object that submits a probe concerning a message containing an IPM or IPN shall specify as the length of the message's content the size in octets of the encoding of the instance in question of the InformationObject of Section 2 (a choice of an IPM or an IPN) when the basic encoding rules of Recommendation X.209 are followed. If those rules permit several (e.g., both primitive and constructed) encodings of that InformationObject, the content length may reflect any one of them.

20.4 *Encoded information types*

A secondary object that submits a message containing an IPM or IPN shall specify the basic encoded information type (EITs) and non-basic parameters (NBPs) of the message as follows.

In the case of an IPN, the basic EITs shall be *unspecified.*

In the case of an IPM, the basic EITs and NBPs shall be specified in accordance with the following rules:

a) *Multiple body parts:* The basic EITs (if any) and NBPs (if any) of the message shall comprise the logical union of the basic EITs and NBPs of the IPM's individual body parts, respectively.

b) (Forwarded) *message body part:* The basic EITs (if any) and NBPs (if any) of a message body part shall be those of the forwarded message.

c) *Externally defined body part:* An externally defined body part whose extended type corresponds to a basic type (see Annex B) shall be treated in the manner prescribed for the basic type.

Any other extended body part shall be handled as follows. If there corresponds to the type one or more externally defined EITs, they shall be specified. Otherwise, the *undefined* EIT shall be indicatd. In either case, no NBPs shall be specified.

d) *Basic body part:* The basic EITs (if any) and NBPs (if any) of an individual body part of type other than message and externally defined shall depend upon that body part type as specified in Table 2/X.420. A body part type for which the table specifies no basic EITs shall result in the setting of no bits in the basic EITs bit string.

e) *Encrypted body part*: The effect of an encrypted body part upon the basic EITs and NBPs to be specified is for further study.

TABLE 2/X.420

Interpersonal messaging basic EITs and NBPs

Body part type	Basic EIT	NBPs
IA5 text	IA5 text	–
Voice	Voice	–
G3 facsimile	G3 facsimile	G3 facsimile
G4 class 1	G4 class 1	G4 class 1/mixed-mode
Teletex	Teletex	Teletex
Videotex	Videotex	–
Encrypted		
Message	(see text)	(see text)
Mixed-mode	Mixed-mode	G4 class 1/mixed-mode
Bilaterally defined	Undefined	–
Nationally defined	Undefined	–
Externally defined	(see text)	(see text)

21 Port realization

How an MS or the MTS concretely realizes the secondary ports it supplies is specified in Recommendation X.419.

How a UA, TLMA, or AU concretely realizes the primary ports it supplies is beyond the scope of this Recommendation.

Note 1 – A UA's user interface is a local matter. A wide variety of interfaces involving, e.g., a wide variety of input/output devices are possible.

Note 2 – A TLMA's realization of its primary ports is specified by Recommendation T.330.

Note 3 – An AU provides its primary ports by means of the particular communication system to which that AU provides access.

22 Conformance

The requirements a secondary object (excluding the MTS) and its implementor shall meet when the latter claims the former's conformance to this Recommendation are identified below. A number of the conformance requirements distinguish between *support upon origination* and *support upon reception.*

22.1 *Origination versus reception*

A UA, TLMA or AU shall be said to **support upon origination** a particular heading field, heading extension, basic body part type, or extended body part type if, and only if, it accepts, preserves, and emits, in full accord with this Recommendation, that particular heading field or extension, or body parts of that particular basic or extended type, whenever a user calls upon it to convey an IPM containing them to the MTS or the user's MS (the latter only in the case of a UA).

A UA, TLMA, or AU shall be said to **support upon reception** a particular heading field, heading extension, basic body part type, or extended body part type if, and only if, it accepts, preserves, and emits, in full accord with this Recommendation, that particular heading field or extension, or body parts of that particular basic or extended type, whenever the MTS or a user's MS (the latter only in the case of a UA) calls upon it to convey to the user an IPM containing them.

Note – In point of fact, a PDAU supports nothing upon origination because it is not a supplier of the origination port.

22.2 *Statement requirements*

The implementor of an IPMS, UA, IPMS MS, TLMA, or AU shall state the following. For each item below he shall make separate statements concerning conformance upon origination and conformance upon reception:

a) The heading fields and heading extensions for which he claims conformance.

b) The basic and extended body part types for which he claims conformance.

c) In the case of an IPMS UA or IPMS MS, the interpersonal messaging-specific MS attributes for which it claims conformance.

22.3 *Static requirements*

An IPMS UA, IPMS MS, TLMA or AU shall satisfy the following static requirements:

a) An IPMS UA, IPMS MS, TLMA, or AU shall implement the heading fields and heading extensions, and the basic and extended body part types for which conformance is claimed.

b) An IPMS UA or IPMS MS shall support the interpersonal messaging-specific MS attributes for which conformance is claimed, but including as a minimum those designated mandatory in Annex C.

c) An IPMS UA, IPMS MS, TLMA, or AU shall concretely realize its abstract ports as specified in § 21.

d) An IPMS UA or IPMS MS shall be able to both submit and accept delivery of messages of both of the content types of § 20.2. A TLMA or AU shall be able to both import and export such messages.

22.4 *Dynamic requirements*

An IPMS UA, IPMS MS, TLMA, or AU shall satisfy the following dynamic requirements:

a) An IPMS UA or IPMS MS shall follow the rules of operation specified in § 18 or § 19, respectively.

b) An IPMS UA, IPMS MS, TLMA, or AU shall submit and accept delivery of messages whose contents are as specified in § 20.

c) An IPMS UA, IPMS MS, TLMA, or AU shall register with the MTS its ability to accept delivery of messages of both of the content types of § 20.2.

ANNEX A

(to Recommendation X.420)

Heading extensions

This Annex is an integral part of this Recommendation.

This Annex defines all (presently defined) heading extensions.

A.1 *Incomplete copy*

The **incomplete copy** heading extension, by its presence, indicates that one of more body parts or heading fields are absent from the body of (the present instance of) the IPM. The extension comprises a null (by default).

Incomplete-copy HEADING-EXTENSION
::= id-hex-incomplete copy

If this extension is absent from the extension heading field, all body parts shall be considered present.

A.2 *Languages*

The **languages** heading extension identifies the languages used in the composition of the IPM's subject heading field and body. The extension comprises a set of zero or more printable strings, each one of the two-character language codes identified by ISO 639.2.

```
languages HEADING-EXTENSION
        VALUE SET OF Language
        ::= id-hex-languages

Language ::= PrintableString (SIZE (2. .2))
```

If this extension is absent from the extensions heading field or no languages are indicated, the languages shall be considered unspecified.

ANNEX B

(to Recommendation X.420)

Extended body part types

This Annex is an integral part of this Recommendation.

For each basic body part type, this Recommendation defines as follows an equivalent extended body part type.

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

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

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

g4-class1-body-part EXTENDED-BODY-PART-TYPE
        DATA        G4ClassBodyPart
        ::= id-et-g4-class1

teletex-body-part EXTENDED-BODY-PART-TYPE
        PARAMETERS  TeletexParameters IDENTIFIED BY id-ep-teletex
        DATA        TeletexData
        ::= id-et-teletex

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

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

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

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

bilaterally-defined-body-part EXTENDED-BODY-PART-TYPE
        DATA            BilaterallyDefinedBodyPart
        ::= id-et-bilaterally-defined

nationally-defined-body-part EXTENDED-BODY-PART-TYPE
        DATA            NationallyDefinedBodyPart
        ::= id-et-nationally-defined
```

ANNEX C

(to Recommendation X.420)

Message store attributes

This Annex is an integral part of this Recommendation.

As described in Recommendation X.413, an MS maintains and provides access to certain attributes (e.g., the importance) of each information object it holds. An attribute comprises a type and, depending upon the type, one or more values. Attributes that may assume several values simultaneously (all pertaining to one object) are termed multi-valued; those that may assume just one value, single-valued. Some attributes pertain to information objects of all kinds, others only to those of certain kinds (e.g., those of Section 2).

This Annex defines the MS attributes specific to interpersonal messaging.

All of the attributes defined in this Annex, except those corresponding to extended body part types (which cannot be enumerated; see § C.3.6), are listed alphabetically, for reference, in the first column of Table C-1/X.420. This Table records their presence in a delivered message entry. None of them appears in either a delivered report entry or a returned content entry. See Recommendation X.413 for an elaboration of the Table's legend.

C.1 *Summary attributes*

Some attributes summarize an interpersonal messaging information object. These attributes are defined and described below.

C.1.1 *IPM entry type*

The **IPM entry type** attribute identifies an information object's type.

```
ipm-entry-type ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX IPMEntryType
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-sat-ipm-entry-type

IPMEntryType ::= ENUMERATED {
        ipm(0),
        rn(1),
        nrn(2) }
```

This attribute may assume any one of the following values:

a) *ipm:* The information object is an IPN.

b) *rn:* The information object is an RN.

c) *nrn:* The information object is an NRN.

An MS that supports this attribute shall maintain it for an information object that it holds if, and only if, that object is a message whose content is an IPM or IPN.

TABLE C-1/X.420

Summary of MS attributes

Attribute	V	L	P IPM	P NRN	P RN	L	S
Acknowledgement mode	S	O	–	–	M	Y	Y
Authorizing users	M	O	C	–	–	Y	N
Auto-forward comment	S	O	–	C	–	Y	N
Auto-forwarded	S	O	C	–	–	Y	Y
Bilaterally defined body parts	M	O	C	–	–	N	N
Blind copy recipients	M	O	C	–	–	Y	N
Body	S	M	M	–	–	N	N
Conversion EITs	M	O	–	C	C	Y	N
Copy recipients	M	O	C	–	–	Y	N
Discard reason	S	O	–	C	–	Y	Y
Encrypted body parts	M	O	C	–	–	N	N
Encrypted data	M	O	C	–	–	N	N
Encrypted parameters	M	O	C	–	–	N	N
Expiry time	S	O	C	–	–	Y	N
Extended body part types	M	O	C	–	–	Y	Y
G3 facsimile body parts	M	O	C	–	–	N	N
G3 facsimile data	M	O	C	–	–	N	N
G3 facsimile parameters	M	O	C	–	–	N	N
G4 class 1 body parts	M	O	C	–	–	N	N
Heading	S	M	M	–	–	N	N
IA5 text body parts	M	O	C	–	–	N	N
IA5 text data	M	O	C	–	–	N	N
IA5 text parameters	M	O	C	–	–	N	N
Importance	S	O	C	–	–	Y	Y
Incomplete copy	S	O	C	–	–	Y	N
IPM entry type	S	M	M	M	M	Y	Y
IPM preferred recipient	S	O	–	C	C	Y	N
IPM synopsis	S	O	M	–	–	N	N
IPM originator	S	O	–	C	C	Y	N
Languages	M	O	C	–	–	Y	N
Message body parts	M	O	C	–	–	N	N
Message data	M	O	C	–	–	N	N
Message parameters	M	O	C	–	–	N	N
Mixed-mode body parts	M	O	C	–	–	N	N
Nationally defined body parts	M	O	C	–	–	N	N
Non-receipt reason	S	O	–	M	–	Y	Y
NRN requestors	M	O	C	–	–	Y	N
Obsoleted IPMs	M	O	C	–	–	Y	N
Originator	S	O	C	–	–	Y	N

TABLE C-1/X.420 *(cont.)*

Attribute	V	L	P IPM	P NRN	P RN	L	S
Primary recipients	M	O	C	–	–	Y	N
Receipt time	S	O	–	–	M	Y	N
Related IPMs	M	O	C	–	–	Y	N
Replied-to IPM	S	O	C	–	–	Y	N
Reply recipients	M	O	C	–	–	Y	N
Reply requestors	M	O	C	–	–	Y	N
Reply time	S	O	C	–	–	Y	N
Returned IPM	S	O	–	C	–	Y	N
RN requestors	M	O	C	–	–	Y	N
Sensitivity	S	O	C	–	–	Y	Y
Subject	S	O	C	–	–	Y	N
Subject IPM	S	M	–	M	M	Y	N
Suppl. receipt info	S	O	–	–	C	Y	N
Teletex body parts	M	O	C	–	–	N	N
Teletex data	M	O	C	–	–	N	N
Teletex parameters	M	O	C	–	–	N	N
This IPM	S	M	M	–	–	Y	N
Videotex body parts	M	O	C	–	–	N	N
Videotex data	M	O	C	–	–	N	N
Videotex parameters	M	O	C	–	–	N	N
Voice body parts	M	O	C	–	–	N	N
Voice data	M	O	C	–	–	N	N
Voice parameters	M	O	C	–	–	N	N

V Single/multi valued

L Support level by MS and access UA

P Presence in delivered message entry

L Available for List, alert

S Available for summarize

C.1.2 *IPM synopsis*

The **IPM synopsis** attribute gives the structure, characteristics, size and processing status of an IPM at the granularity of individual body parts.

```
ipm-synopsis ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX IPMSynopsis
        SINGLE VALUE
        ::= id-sat-ipm-synopsis
```

The synopsis of an IPM comprises a synopsis of each of its body parts. The synopses appear in the order in which the body parts appear.

```
IPMSynopsis ::= SEQUENCE OF BodyPartSynopsis
```

The synopsis of a body part takes either of two forms depending upon whether the body part is of type message. This enables the synopsis of a forwarding IPM to encompass the body parts of each forwarded IPM (recursively), as well as those of the forwarding IPM itself.

```
BodyPartSynopsis ::= CHOICE {
        message         [0] MessageBodyPartSynopsis
        non-message     [1] NonMessageBodyPartSynopsis }

MessageBodyPartSynopsis ::= SEQUENCE {
        number          [0] SequenceNumber,
        synopsis        [1] IPMSynopsis }

NonMessageBodyPartSynopsis ::= SEQUENCE {
        type            [0] OBJECT IDENTIFIER,
        parameters      [1] ExternallyDefinedParameters,
        size            [2] INTEGER,
        processed       [3] BOOLEAN DEFAULT FALSE }
```

The synopsis of a message body part has the following components:

a) **Number** (M): The sequence number that the MS assigns to the entry that the message body part represents.

b) **Synopsis** (M): The synopsis of the IPM that forms the content of the message that the body part represents.

The synopsis of a body part of type other than message has the following components. For purposes of this synopsis, the body part is considered to be of type externally defined, whether or not (see Annex B) it was so conveyed to the MS:

a) **Type** (M): The body part's extended type, i.e., the direct-reference component of the body part's data component. An object identifier.

b) **Parameters** (M): The body part's format and control parameters, i.e., the body part's parameters component. An Any.

c) **Size** (M): The size in octets of the encoding of the Encoding component of the body part's Data component when the basic encoding rules of Recommendation X.209 are followed. If those rules permit several (e.g., both primitive and constructed) encodings of the component, the size may reflect any one of them. An Integer.

d) **Processed** (D *false*): An indication of whether or not the body part has been conveyed to the UA by means of the MS' list or fetch abstract operation. A Boolean.

An MS that supports this attribute shall maintain it for an information object that it holds if, and only if, that object is a message whose content is an IPM.

Note – As a consequence of its variability, the value of the size component should be considered only an estimate of the body part's size.

C.2 *Heading attributes*

Some attributes are derived from the heading of an IPM. These attributes are defined and described below.

C.2.1 *Heading*

The **heading** attribute is the (entire) heading of an IPM.

```
heading ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX Heading
        SINGLE VALUE
        ::= id-hat-heading
```

An MS that supports this attribute shall maintain it for an information object that it holds if, and only if, that object is a message whose content is an IPM.

C.2.2 *Heading analyses*

Some attributes have as their values O/R descriptors selected after analysis of the heading. They identify the "primary", "copy", and blind "copy" recipients of an IPM of whom an RN, NRN, or reply is requested.

```
rn-requestors ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ORDescriptor
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-hat-rn-requestors
```

```
nrn-requestors ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ORDescriptor
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-hat-nrn-requestors

reply-requestors ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ORDescriptor
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-hat-reply-requestors
```

An MS that supports one of these attributes shall maintain it for an information object that it holds if, and only if, that object is a message whose content is an IPM whose heading requests, of at least one user, or DL, an RN, NRN, or reply, respectively. It shall maintain one attribute value for every recipient specifier in the IPM's primary, copy, or blind copy recipients field whose notification-requests component includes the value rn (in the case of the first attribute) or nrn (in the case of the second, or whose Reply-requested component signifies, by either its presence or its absence, that a reply is requested (in the case of the third). The value shall be the recipient's specifier's recipient component.

C.2.3 *Heading fields*

Some attributes bear the names of heading fields and have those fields as their values. The ordering for the expiry time and reply time attributes is increasing chronological order.

```
this-ipm ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX This IPMField
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-hat-this-ipm

originator ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX OriginatorField
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-hat-originator

replied-to-IPM ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX RepliedToIPMField
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-hat-replied-to-IPM

subject ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX SubjectField
        MATCHES FOR EQUALITY SUBSTRINGS
        SINGLE VALUE
        ::= id-hat-subject

expiry-time ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ExpiryTimeField
        MATCHES FOR EQUALITY ORDERING
        SINGLE VALUE
        ::= id-hat-expiry-time

reply-time ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ReplyTimeField
        MATCHES FOR EQUALITY ORDERING
        SINGLE VALUE
        ::= id-hat-reply-time

importance ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ImportanceField
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-hat-importance
```

```
sensitivity ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX SensitivityField
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-hat-sensitivity

auto-forwarded ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX AutoForwardedField
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-hat-auto-forwarded
```

An MS that supports one of these attributes shall maintain it for an information object that it holds if, and only if, that object is a message whose content is an IPM whose heading contains the field whose name the attribute bears.

C.2.4 *Heading sub-fields*

Some attributes bear the names of heading fields and have sub-fields of those fields as their values.

```
authorizing-users ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX AuthorizingUsersSubfield
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-hat-authorizing-users

primary-recipients; ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX PrimaryRecipientsSubField
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-hat-primary-recipients

copy-recipients ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX CopyRecipientsSubfield
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-hat-copy-recipients

blind-copy-recipients ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX BlindCopyRecipientsSubfield
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-hat-blind-copy-recipients

obsoleted-IPMs ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ObsoletedIPMsSubfield
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-hat-obsoleted-IPMs

related-IPMs ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX RelatedIPMsSubfield
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-hat-related-IPMs

reply-recipients ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ReplyRecipientsSubfield
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-hat-reply-recipients
```

An MS that supports one of these attributes shall maintain it for an information object that it holds if, and only if, that object is a message whose content is an IPM whose heading contains the field whose name the attribute bears. It shall maintain one attribute value for each sub-field.

C.2.5 *Heading extensions*

Some attributes bear the names of heading extensions and have as their values the values of those extensions or a part thereof.

```
incomplete-copy ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX incompleteCopyExtensionValue
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-hat-incomplete-copy

languages ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX Language
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-hat-languages
```

An MS that supports one of these attributes shall maintain it for an information object that it holds if, and only if, that object is a message whose content is an IPM whose heading contains the extension whose name the attribute bears. In the case of the language attribute, the MS shall maintain one attribute value for each language the extension identifies.

C.3 *Body attributes*

Some attributes are derived from the body of an IPM. These attributes are defined and described below.

C.3.1 *Body*

The **body** attribute is the (entire) body of an IPM.

```
body ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX Body
        SINGLE VALUE
        ::= id-bat-body
```

An MS that supports this attribute shall maintain it for an information object that it holds if, and only if, that object is a message whose content is an IPM.

C.3.2 *Basic body parts*

Some attributes bear the names of basic body part types and have, with one exception, such body parts as their values.

An MS holds each forwarded IPM (i.e., each message body part) as an information object in its own right, separate from the forwarding IPM. That information object, of course, is a message whose content is an IPM. The message body parts attribute below, therefore, has as its values the sequence numbers the MS assigns to these messages.

```
ia5-text-body-parts ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX IA5TextBodyPart
        MULTI VALUE
        ::= id-bat-ia5-text-body-parts

voice-body-parts ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX VoiceBodyPart
        MULTI VALUE
        ::= id-bat-voice-body-parts

g3-facsimile-body-parts ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX G3FacsimileBodyPart
        MULTI VALUE
        ::= id-bat-g3-facsimile-body-parts

g4-class1-body-parts ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX G4Class1BodyPart
        MULTI VALUE
        ::= id-bat-g4-class1-body-parts

teletex-body-parts ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX TeletexBodyPart
        MULTI VALUE
        ::= id-bat-teletex-body-parts
```

videotex-body-parts ATTRIBUTE
WITH ATTRIBUTE-SYNTAX VideotexBodyPart
MULTI VALUE
::= id-bat-videotex-body-parts

encrypted-body-parts ATTRIBUTE
WITH ATTRIBUTE-SYNTAX EncryptedBodyPart
MULTI VALUE
::= id-bat-encrypted-body-parts

message-body-parts ATTRIBUTE
WITH ATTRIBUTE-SYNTAX SequenceNumber
MULTI VALUE
::= id-bat-message-body-parts

mixed-mode-parts ATTRIBUTE
WITH ATTRIBUTE-SYNTAX MixedModeBodyPart
MULTI VALUE
::= id-bat-mixed-mode-body-parts

bilaterally-defined-body-parts ATTRIBUTE
WITH ATTRIBUTE-SYNTAX BilaterallyDefinedBodyPart
MULTI VALUE
::= id-bat-bilaterally-defined-body-parts

nationally-defined-body-parts ATTRIBUTE
WITH ATTRIBUTE-SYNTAX NationallyDefinedBodyPart
MULTI VALUE
::= id-bat-nationally-defined-body-parts

An MS that supports one of these attributes shall maintain it for an information object that it holds if, and only if, that object is a message whose content is an IPM whose body contains one or more body parts of the type whose name the attribute bears. It shall maintain one attribute value for each such body part.

C.3.3 *Basic body part parameters components*

Some attributes bear the names of basic body part types and have the parameters components of such body parts as their values.

ia5-text-parameters ATTRIBUTE
WITH ATTRIBUTE-SYNTAX IA5TextParameters
MULTI VALUE
::= id-bat-ia5-text-parameters

voice-text-parameters ATTRIBUTE
WITH ATTRIBUTE-SYNTAX IA5VoiceParameters
MULTI VALUE
::= id-bat-voice-parameters

g3-facsimile-parameters ATTRIBUTE
WITH ATTRIBUTE-SYNTAX G3FacsimileParameters
MULTI VALUE
::= id-bat-g3-facsimile-parameters

teletex-parameters ATTRIBUTE
WITH ATTRIBUTE-SYNTAX TeletexParameters
MULTI VALUE
::= id-bat-teletex-parameters

videotex-parameters ATTRIBUTE
WITH ATTRIBUTE-SYNTAX VideotexParameters
MULTI VALUE
::= id-bat-videotex-parameters

encrypted-parameters ATTRIBUTE
WITH ATTRIBUTE-SYNTAX EncryptedParameters
MULTI VALUE
::= id-bat-encrypted-parameters

message-parameters ATTRIBUTE
WITH ATTRIBUTE-SYNTAX MessageParameters
MULTI VALUE
::= id-bat-message-parameters

An MS that supports one of these attributes shall maintain it for an information object that it holds if, and only if, that object is a message whose content is an IPM whose body contains one or more body parts of the type whose name the attribute bears. It shall maintain one attribute value for each such body part.

C.3.4 *Basic body part data components*

Some attributes bear the names of basic body part types and have the data components of such body parts as their values.

```
ia5-text-data ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX IA5TextData
        MULTI VALUE
        ::= id-bat-ia5-text-data

voice-data ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX VoiceData
        MULTI VALUE
        ::= id-bat-voice-data

g3-facsimile-data ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX G3FacsimileData
        MULTI VALUE
        ::= id-bat-g3-facsimile-data

teletex-data ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX TeletexData
        MULTI VALUE
        ::= id-bat-teletex-data

videotex-data ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX VideotexData
        MULTI VALUE
        ::= id-bat-videotex-data

encrypted-data ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX EncryptedData
        MULTI VALUE
        ::= id-bat-encrypted-data

message-data ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MessageData
        MULTI VALUE
        ::= id-bat-message-data
```

An MS that supports one of these attributes shall maintain it for an information object that it holds if, and only if, that object is a message whose content is an IPM whose body contains one or more body parts of the type whose name the attribute bears. It shall maintain one attribute for each such body part.

C.3.5 *Extended body part types*

The **extended body part types** attribute identifies the extended body part types represented in an IPM.

```
extended-body-parts ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX OBJECT IDENTIFIER
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-bat-extended-body-parts
```

An MS that supports this attribute shall maintain it for an information object that it holds if, and only if, that object is a message whose content is an IPM whose body contains one or more externally defined body parts. It shall maintain one attribute value for every such type present. The value shall denote the type as specified in § 7.3.12.

Note – Each value of this attribute corresponds to one of the attributes described in § C.3.6 below.

C.3.6 *Extended body parts*

Some attributes, unnamed, have as their values the encoding components (see § 7.3.12) of the ASN.1 externals that constitute the data components of externally defined body parts.

To each extended body part type there correspond two attributes. The first attribute is denoted by the object identifier that is the direct-reference component (again, see § 7.3.12) of the external that constitutes the data component of a body part of that type. The content of this first attribute is that data component. The second attribute is denoted by the object identifier that is the direct-reference component of the external that constitutes the parameters component of a body part of that type. The content of this second attribute is that parameters component.

An MS that supports one of these body parts shall maintain both attributes for an information object that it holds if, and only if, that object is a message whose content is an IPM whose body contains one or more body parts of the type that corresponds to that attribute. It shall maintain one value of each attribute for each such body part.

Note 1 – The extended body part attributes cannot be enumerated in practice because the extended body part types cannot be so enumerated.

Note 2 – The extended body part types attribute (see § C.3.5) determines the extended body part attributes for a particular IPM.

C.4 *Notification attributes*

Some attributes are derived from an IPN. These attributes are defined and described below.

C.4.1 *Common fields*

Some attributes that bear the names of common fields and have those fields as their values.

```
subject-ipm ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX SubjectIPMField
        MATCHES FOR EQUALITY SUBSTRINGS
        SINGLE VALUE
        ::= id-nat-subject-ipm

ipn-originator ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX IPNOriginatorField
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-nat-ipn-originator

ipn-preferred-recipient ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX IPMPreferredRecipientField
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-nat-ipm-preferred-recipient

conversion-eits ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MS-EITs
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-nat-conversion-eits
```

An MS that supports one of these attributes shall maintain it for an information object that it holds if, and only if, that object is a message whose content is an IPN that contains the field whose name the attribute bears.

C.4.2 *Non-receipt fields*

Some attributes bear the names of non-receipt fields and have those fields as their values.

```
non-receipt-reason ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX NonReceiptReasonField
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-nat-non-receipt-reason
```

discard-reason ATTRIBUTE
WITH ATTRIBUTE-SYNTAX DiscardReasonField
MATCHES FOR EQUALITY
SINGLE VALUE
::= id-nat-discard-reason

auto-forward-comment ATTRIBUTE
WITH ATTRIBUTE-SYNTAX AutoForwardCommentField
MATCHES FOR EQUALITY SUBSTRINGS
SINGLE VALUE
::= id-nat-auto-forward-comment

returned-IPM ATTRIBUTE
WITH ATTRIBUTE-SYNTAX ReturnedIPMField
SINGLE VALUE
::= id-nat-returned-ipm

An MS that supports one of these attributes shall maintain it for an information that it holds if, and only if, that object is a message whose content is an NRN that contains the field whose name the attribute bears.

C.4.3 *Receipt fields*

Some attributes bear the names of receipt fields and have those fields as their values. The ordering for the receipt time attribute is increasing chronological order.

receipt-time ATTRIBUTE
WITH ATTRIBUTE-SYNTAX ReceiptTimeField
MATCHES FOR EQUALITY ORDERING
SINGLE VALUE
::= id-nat-receipt-time

acknowledgment ATTRIBUTE
WITH ATTRIBUTE-SYNTAX AcknowledgmentModeField
MATCHES FOR EQUALITY
SINGLE VALUE
::= id-nat-acknowledgment-mode

suppl-receipt-info ATTRIBUTE
WITH ATTRIBUTE-SYNTAX SupplReceiptInfoField
MATCHES FOR EQUALITY SUBSTRINGS
SINGLE VALUE
::= id-nat-suppl-receipt-info

An MS that supports one of these attributes shall maintain it for an information object that it holds if, and only if, that object is a message whose content is an RN that contains the field whose name the attribute bears.

ANNEX D

(to Recommendation X.420)

Reference definition of object identifiers

This Annex is an integral part of this Recommendation.

This Annex defines for reference purposes various object identifiers cited in the ASN.1 modules of subsequent Annexes. It uses ASN.1.

All object identifiers this Recommendation assigns are assigned in this Annex. The Annex is definitive for all but those for ASN.1 modules and the IPMS application itself. The definitive assignments for the former occur in the modules themselves; other references to them appear in IMPORT clauses. The latter is fixed.

```
IMPSObjectIdentifiers {joint-iso-ccitt
      mhs-motis(6) ipms(1) modules(0) object-identifiers(0)}
DEFINITIONS IMPLICIT TAGS ::=
BEGIN


-- Prologue
-- Exports everything.

IMPORTS -- nothing --;

ID ::= OBJECT IDENTIFIER


-- Interpersonal Messaging (not definitive)

id-ipms ID ::= {joint-iso-ccitt mhs-motis(6) ipms(1)} - -not definitive


-- Categories

id-mod  ID ::= { id-ipms  0 } --  modules; not definitive
id-ot   ID ::= { id-ipms  1 } --  object types
id-pt   ID ::= { id-ipms  2 } --  port types
id-ref  ID ::= { id-ipms  3 } --  refinements
id-et   ID ::= { id-ipms  4 } --  extended body part types
id-hex  ID ::= { id-ipms  5 } --  heading extensions
id-sat  ID ::= { id-ipms  6 } --  summary attributes
id-hat  ID ::= { id-ipms  7 } --  heading attributes
id-bat  ID ::= { id-ipms  8 } --  body attributes
id-nat  ID ::= { id-ipms  9 } --  notification attributes
id-mct  ID ::= { id-ipms 10 } --  message content types
id-ep   ID ::= { id-ipms 11 } --  extended body part parameters


-- Modules

id-mod-object-identifiers          ID ::= { id-mod  0 } -- not definitive
id-mod-functional-objects          ID ::= { id-mod  1 } -- not definitive
id-mod-information-objects         ID ::= { id-mod  2 } -- not definitive
id-mod-abstract-service            ID ::= { id-mod  3 } -- not definitive
id-mod-heading-extensions          ID ::= { id-mod  6 } -- not definitive
id-mod-extended-body-part-types    ID ::= { id-mod  7 } -- not definitive
id-mod-message-store-attributes    ID ::= { id-mod  8 } -- not definitive
id-mod-upper-bounds                ID ::= { id-mod 10 } -- not definitive


-- Object types

id-ot-ipme                         ID ::= { id-ot 0 }
id-ot-ipms-user                    ID ::= { id-ot 1 }
id-ot-ipms                         ID ::= { id-ot 2 }
id-ot-ipms-ua                      ID ::= { id-ot 3 }
id-ot-ipms-ms                      ID ::= { id-ot 4 }
id-ot-tlma                         ID ::= { id-ot 5 }
id-ot-tlxau                        ID ::= { id-ot 6 }
id-ot-pdau                         ID ::= { id-ot 7 }


-- Port types

id-pt-origination                  ID ::= { id-pt 0 }
id-pt-reception                    ID ::= { id-pt 1 }
id-pt-management                   ID ::= { id-pt 2 }
```

-- *Refinements*

```
id-ref-primary                  ID ::= { id-ref 0 }
id-ref-secondary                ID ::= { id-ref 1 }
```

-- *Extended body part types*

```
id-et-ia5-text                  ID ::= { id-et 0 }
id-et-voice                     ID ::= { id-et 1 }
id-et-g3-facsimile              ID ::= { id-et 2 }
id-et-g4-class1                 ID ::= { id-et 3 }
id-et-teletex                   ID ::= { id-et 4 }
id-et-videotex                  ID ::= { id-et 5 }
id-et-encrypted                 ID ::= { id-et 6 }
id-et-message                   ID ::= { id-et 7 }
id-et-mixed-mode                ID ::= { id-et 8 }
id-et-bilaterally-defined       ID ::= { id-et 9 }
id-et-nationally-defined        ID ::= { id-et 10 }
```

-- *Heading extensions*

```
id-hex-incomplete-copy          ID ::= { id-hex 0 }
id-hex-languages                ID ::= { id-hex 1 }
```

-- *Summary attributes*

```
id-sat-ipm-entry-type           ID ::= { id-sat 0 }
id-sat-ipm-synopsis             ID ::= { id-sat 1 }
```

-- *Heading attributes*

```
id-hat-heading                  ID ::= { id-hat 0 }
id-hat-this-ipm                 ID ::= { id-hat 1 }
id-hat-originator               ID ::= { id-hat 2 }
id-hat-replied-to-IPM           ID ::= { id-hat 3 }
id-hat-subject                  ID ::= { id-hat 4 }
id-hat-expiry-time              ID ::= { id-hat 5 }
id-hat-reply-time               ID ::= { id-hat 6 }
id-hat-importance               ID ::= { id-hat 7 }
id-hat-sensitivity              ID ::= { id-hat 8 }
id-hat-auto-forwarded           ID ::= { id-hat 9 }
id-hat-authorizing-users        ID ::= { id-hat 10 }
id-hat-primary-recipients       ID ::= { id-hat 11 }
id-hat-copy-recipients          ID ::= { id-hat 12 }
id-hat-blind-copy-recipients    ID ::= { id-hat 13 }
id-hat-obsoleted-IPMs           ID ::= { id-hat 14 }
id-hat-related-IPMs             ID ::= { id-hat 15 }
id-hat-reply-recipients         ID ::= { id-hat 16 }
id-hat-incomplete-copy          ID ::= { id-hat 17 }
id-hat-languages                ID ::= { id-hat 18 }
id-hat-rn-requestors            ID ::= { id-hat 19 }
id-hat-nrn-requestors           ID ::= { id-hat 20 }
id-hat-reply-requestors         ID ::= { id-hat 21 }
```

-- *Body attributes*

```
id-bat-body                          ID ::= { id-bat 0 }
id-bat-ia5-text-body-parts           ID ::= { id-bat 1 }
id-bat-voice-body-parts              ID ::= { id-bat 2 }
id-bat-g3-facsimile-body-parts       ID ::= { id-bat 3 }
id-bat-g4-class1-body-parts          ID ::= { id-bat 4 }
id-bat-teletex-body-parts            ID ::= { id-bat 5 }
id-bat-videotex-body-parts           ID ::= { id-bat 6 }
id-bat-encrypted-body-parts          ID ::= { id-bat 7 }
id-bat-message-body-parts            ID ::= { id-bat 8 }
id-bat-mixed-mode-body-parts         ID ::= { id-bat 9 }
id-bat-bilaterally-defined-body-parts ID ::= { id-bat 10 }
id-bat-nationally-defined-body-parts ID ::= { id-bat 11 }
id-bat-extended-body-part-types      ID ::= { id-bat 12 }
id-bat-ia5-text-parameters           ID ::= { id-bat 13 }
id-bat-voice-parameters              ID ::= { id-bat 14 }
id-bat-g3-facsimile-parameters       ID ::= { id-bat 15 }
id-bat-teletex-parameters            ID ::= { id-bat 16 }
id-bat-videotex-parameters           ID ::= { id-bat 17 }
id-bat-encrypted-parameters          ID ::= { id-bat 18 }
id-bat-message-parameters            ID ::= { id-bat 19 }
id-bat-ia5-text-data                 ID ::= { id-bat 20 }
id-bat-voice-data                    ID ::= { id-bat 21 }
id-bat-g3-facsimile-data             ID ::= { id-bat 22 }
id-bat-teletex-data                  ID ::= { id-bat 23 }
id-bat-videotex-data                 ID ::= { id-bat 24 }
id-bat-encrypted-data                ID ::= { id-bat 25 }
id-bat-message-data                  ID ::= { id-bat 26 }
```

-- *Notification attributes*

```
id-nat-subject-ipm                   ID ::= { id-nat 0 }
id-nat-ipn-originator                ID ::= { id-nat 1 }
id-nat-ipm-preferred-recipient       ID ::= { id-nat 2 }
id-nat-conversion-eits               ID ::= { id-nat 3 }
id-nat-non-receipt-reason            ID ::= { id-nat 4 }
id-nat-discard-reason                ID ::= { id-nat 5 }
id-nat-auto-forward-comment          ID ::= { id-nat 6 }
id-nat-returned-ipm                  ID ::= { id-nat 7 }
id-nat-receipt-time                  ID ::= { id-nat 8 }
id-nat-acknowledgment-mode           ID ::= { id-nat 9 }
id-nat-suppl-receipt-info            ID ::= { id-nat 10 }
```

-- *Message content types (for use by MS only)*

```
id-mct-p2-1984 ID ::= { id-mct 0 } -- P2 1984
id-mct-p2-1988 ID ::= { id-mct 1 } -- P2 1988
```

-- *Extended body part parameters*

```
id-ep-ia5-text                       ID ::= { id-ep 0 }
id-ep-voice                          ID ::= { id-ep 1 }
id-ep-g3-facsimile                   ID ::= { id-ep 2 }
id-ep-teletex                        ID ::= { id-ep 4 }
id-ep-videotex                       ID ::= { id-ep 5 }
id-ep-encrypted                      ID ::= { id-ep 6 }
id-ep-message                        ID ::= { id-ep 7 }
```

END -- *of IPMSObjectIdentifiers*

ANNEX E

(to Recommendation X.420)

Reference definition of abstract information objects

This Annex is an integral part of this Recommendation.

This Annex, a supplement to Section 2, defines for reference purposes the abstract information objects of interpersonal messaging.

```
IPMSInformationObjects {joint-iso-ccitt
    mhs-motis(6) ipms(1) modules(0) information-objects(2) }
DEFINITIONS IMPLICIT TAGS ::=
BEGIN


-- Prologue
-- Exports everything.

IMPORTS
    -- IPMS upper bounds

        ub-auto-forward-comment, ub-free-form-name, ub-ipm-identifier-suffix,
        ub-local-imp-identifier, ub-subject-field, ub-telephone-number
        ----
        FROM IPMSUpperBounds { joint-iso-ccitt
            mhs-motis(6) ipms(1) modules(0) upper-bounds(10) }

    -- DTAM

        ProtocolElement
        ----
        FROM dTAM

    --- MTS abstract service

        EncodedInformationTypes, G3FacsimileNonBasicParameters,
        MessageDeliveryTime, ORAddress, ORName,
        OtherMessageDeliveryFields, SupplementaryInformation,
        TeletexNonBasicParameters,
        ----
        FROM MTSAbstractService { joint-iso-ccitt
            mhs-motis(6) mts(3) modules(0) mts-abstract-service(1) };

Time ::= UTCTime


-- Information object

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


-- IPM

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

```
-- Heading

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


-- Heading component types

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

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

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

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

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

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

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


-- This IPM heading field

This IPMField ::= IPMIdentifier


-- Originator heading field

OriginatorField ::= ORDescriptor


-- Authorizing users heading field

AuthorizingdUsersField ::= SEQUENCE OF AuthorizingUsersSubfield

AuthorizingUsersSubfield ::= ORDescriptor
```

```
-- Primary recipients heading field
PrimaryRecipientsField ::= SEQUENCE OF PrimaryRecipientsSubfield
PrimaryRecipientsSubfield ::= RecipientSpecifier

-- Copy recipients heading field
CopyRecipientsField ::= SEQUENCE OF CopyRecipientsSubfield
CopyRecipientsSubfield ::= RecipientSpecifier

-- Blind copy recipients heading field
BlindCopyRecipientsField ::= SEQUENCE OF BlindCopyRecipientsSubfield
BlindCopyRecipientsSubfield ::= RecipientSpecifier

-- Replied-to IPM heading field
RepliedToIPMField ::= IPMIdentifier

-- Obsoleted IPMs heading field
ObsoletedIPMsField ::= SEQUENCE OF ObsoletedIPMsSubfield
ObsoletedIPMsSubfield ::= IPMIdentifier

-- Related IPMs heading field
RelatedIPMsField ::= SEQUENCE OF RelatedIPMsSubfield
RelatedIPMsSubfield ::= IPMIdentifier

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

-- Expiry time heading field
Expiry-TimeField ::= Time

-- Reply time heading field
ReplyTimeField ::= Time

-- Reply recipient heading field
ReplyRecipientsField ::= SEQUENCE OF ReplyRecipientsSubfield
ReplyRecipientsSubfield ::= ORDescriptor

-- Importance heading field
ImportanceField ::= ENUMERATED {
        low             (0),
        normal          (1),
        high            (2) }
```

```
-- Sensitivity heading field

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


-- Auto-forwarded heading field

AutoForwardedField ::= BOOLEAN


-- Extensions heading field

ExtensionsField ::= SET OF HeadingExtension

HeadingExtension ::= SEQUENCE {
        type    OBJECT IDENTIFIER,
        value   ANY DEFINED BY type DEFAULT NULL NULL }

HEADING-EXTENSION MACRO ::=
BEGIN
                TYPE NOTATION   ::= "VALUE" type | empty
                VALUE NOTATION  ::= value (VALUE OBJECT IDENTIFIER)

END


-- Body

Body ::= SEQUENCE OF BodyPart

BodyPart ::= CHOICE {
        ia5-text                [0]  IA5TextBodyPart,
        voice                   [2]  VoiceBodyPart,
        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,
        externally-defined      [15] ExternallyDefinedBodyPart }


-- IA5 text body part

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

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

IA5TextData ::= IA5String

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


-- Voice body part

VoiceBodyPart ::= SEQUENCE {
        parameters      VoiceParameters,
        data            VoiceData }
```

```
VoiceParameters ::= SET -- for further study
VoiceData ::= BIT STRING -- for further study


-- G3 Facsimile body part
G3FacsimileBodyPart ::= SEQUENCE {
        parameters      G3FacsimileParameters,
        data            G3FacsimileData }
G3FacsimileParameters ::= SET {
        number-of-pages         [0]  INTEGER OPTIONAL,
        non-basic-parameters    [1]  G3FacsimileNonBasicParameters OPTIONAL }
G3FacsimileData ::= SEQUENCE OF BIT STRING


-- G4 class 1 and mixed-mode body parts
G4Class1BodyPart ::= SEQUENCE OF ProtocolElement
MixedModeBodyPart ::= SEQUENCE OF ProtocolElement


-- Telex body part
TeletexBodyPart ::= SEQUENCE {
        parameters      TeletexParameters,
        data            TeletexData }
TeletexParameters ::= SET {
        number-of-pages         [0]  INTEGER OPTIONAL,
        telex-compatible        [1]  BOOLEAN DEFAULT FALSE,
        non-basic-parameters    [2]  TeletexNonBasicParameters OPTIONAL }
TeletexData ::= SEQUENCE OF TeletexString


-- Videotex body part
VideotexBodyPart ::= SEQUENCE {
        parameters      VideotexParameters,
        data            VideotexData }
VideotexParameters ::= SET {
        syntax          [0] VideotexSyntax OPTIONAL
VideotexSyntax ::= INTEGER
        ids                     (0),
        data-syntax1            (1),
        data-syntax2            (2),
        data-syntax3            (3) }
VideotexData ::= VideotexString


-- Encrypted body part
EncryptedBodyPart ::= SEQUENCE {
        parameters      EncryptedParameters,
        data            EncryptedData }
EncryptedParameters ::= SET -- for further study
EncryptedData ::= BIT STRING -- for further study


-- Message body part
MessageBodyPart ::= SEQUENCE {
        parameters      MessageParameters,
        data            MessageData }
```

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

MessageData ::= IPM


-- Bilaterally defined body part

BilaterallyDefinedBodyPart ::= OCTET STRING


-- Nationally defined body part

NationallyDefinedBodyPart ::= ANY


-- Externally defined body part

ExternallyDefinedBodyPart ::= SEQUENCE {
        parameters [0] ExternallyDefinedParameters OPTIONAL,
        data           ExternallyDefinedData }

ExternallyDefinedParameters ::= EXTERNAL

ExternallyDefinedData ::= EXTERNAL

EXTENDED-BODY-PART-TYPE MACRO ::=
BEGIN
        TYPE NOTATION           ::= Parameters Data
        VALUE NOTATION          ::= value (VALUE OBJECT IDENTIFIER)
        Parameters              ::= "PARAMETERS" type "IDENTIFIED" "BY" value (OBJECT
                                    IDENTIFIER) | empty
        Data                    ::= "DATA" type
END

-- IPN

IPN ::= SET {
        -- common-fields -- COMPONENTS OF CommonFields,
        choice [0] CHOICE {
                non-receipt-fields      [0] NonReceiptFields,
                receipt-fields          [1] ReceiptFields } }

RN ::= IPN -- with receipt-fields chosen

NRN ::= IPN -- with non-receipt-fields chosen

CommonFields ::= SET {
        subject-ipn                 SubjectIPNFields,
        ipn-originator          [1] IPNOriginatorField OPTIONAL,
        ipn-preferred-recipient [2] IPNPreferredRecipientField OPTIONAL,
        conversion-eits             ConversionEITsField OPTIONAL }

NonReceiptFields ::= SET {
        non-receipt-reason      [0] NonReceiptReasonField,
        discard-reason          [1] DiscardReasonField OPTIONAL,
        auto-forward-comment    [2] AutoForwardCommentField OPTIONAL,
        returned-ipm            [3] ReturnedIPNField OPTIONAL }

ReceiptFields ::= SET {
        receipt-time            [0] RecipientTimeField,
        acknowledgment-mode     [1] AcknowledgementModeField DEFAULT manual,
        suppl-receipt-info      [2] SupplReceiptInfoField DEFAULT " " }


-- Common fields

SubjectIPMField ::= IPMIdentifier

IPMOriginatorField ::= ORDescriptor

IPMPreferredRecipientField ::= ORDescriptor

ConversionEITsField ::= EncodedInformationTypes
```

```
-- Non-receipt fields

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

DiscardReasonField ::= ENUMERATED {
        ipm-expired                         (0),
        ipm-obsoleted                       (1),
        user-subscription-terminated        (2) }

AutoForwardCommentField ::= AutoForwardComment

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

ReturnedIPMField ::= IPM


-- Receipt fields

ReceiptTimeField ::= Time

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

SupplReceiptInfoField/::= SupplementaryInformation


-- Message store realization

ForwardedInfo ::= SET {
        auto-forwarding-comment [0]
                AutoForwardComment OPTIONAL,
        cover-note [1]
                IA5TextBodyPart OPTIONAL,
        this-ipm-prefix [2]
                PrintableString (SIZE (1. .ub-ipm-identifier-suffix))
                    OPTIONAL }

END  -- of IPMSInformationObjects
```

ANNEX F

(to Recommendation X.420)

Reference definition of functional objects

This Annex is an integral part of this Recommendation.

This Annex, a supplement to §§ 10, 11 and 16, defined for reference purposes the functional objects of interpersonal messaging. It uses the OBJECT and REFINE macros of Recommendation X.407.

```
IPMFunctionalObjects { joint-iso-ccitt
        mhs-motis(6) modules(0) functional-objects(1) }
DEFINITIONS IMPLICIT TAGS ::=
BEGIN
```

```
-- Prologue
-- Exports everything.

IMPORTS
        -- IPMS abstract service
                management, origination, reception
                ----
                FROM IPMSAbstractService { joint-iso-ccitt
                    mhs-motis(6) ipms(1) modules(0) abstract-service(3) }

        -- IPMS object identifiers
                id-ot-ipme, id-ot-ipms, id-ot-ipms-ms, id-ot-ipms-ua,
                id-ot-ipms-user, id-ot-pdau, id-ot-tima, id-ot-tlxau,
                id-ref-primary, id-ref-secondary
                ----
                FROM IPMSObjectIdentifiers { joint-iso-ccitt
                    mhs-motis(6) ipms(1) modules(0) object-identifiers(0) }

         TLMA abstract service
                miscellanea
                ----
                FROM TLMAAbsService { ccitt
                    recommendation(0) t(20) 330 tlmaabsservice(0) }

        -- MS abstract service
                retrieval
                ----
                FROM MSAbstractService { joint-iso-ccitt
                    mhs-motis(6) ms(4) modules(0) abstract-service(1) }

        -- MTS abstract service
                administration, delivery, mTS, submission
                ----
                FROM MTSAbstractService { joint-iso-ccitt
                    mhs-motis(6) mts(3) modules(0) mts-abstract-service(1) }

        -- Abstract service definition conventions
                OBJECT, REFINE
                ----
                FROM AbstractServiceNotation { joint-iso-ccitt
                    mhs-motis(6) asdc(2) modules(0) notation(1) }


-- "Root" object type

ipme OBJECT
        ::= id-ot-ipme


-- Primary refinement

ipme-refinement REFINE ipme AS
        ipms
                origination     [S] PAIRED WITH ipms-user
                reception       [S] PAIRED WITH ipms-user
                management      [S] PAIRED WITH ipms-user
        ipms-user RECURRING
        ::= id-ref-primary


-- Primary object types

ipms-user OBJECT
        PORTS {
                origination     [C],
                reception       [C],
                management      [C] }
        ::= id-ot-ipms-user
```

```
ipms OBJECT
        PORTS }
                origination     [S],
                reception       [S],
                management      [S] }
        ::= id-ot-ipms


-- Secondary refinement

ipms-refinement REFINE ipms AS
        mTS
                submission      [S] PAIRED WITH ipms-ua, ipms-ms
                delivery        [S] PAIRED WITH ipms-ua, ipms-ms
                administration  [S] PAIRED WITH ipms-ua, ipms-ms
        ipms-ua RECURRING
                origination     [S] VISIBLE
                reception       [S] VISIBLE
                management      [S] VISIBLE
        ipms-ms RECURRING
                submission      [S] PAIRED WITH ipms-ua
                retrieval       [S] PAIRED WITH ipms-ua
                administration  [S] PAIRED WITH ipms-ua
        tlma RECURRING
                origination     [S] VISIBLE
                reception       [S] VISIBLE
                management      [S] VISIBLE
        tlxau RECURRING
                origination     [S] VISIBLE
                reception       [S] VISIBLE
                management      [S] VISIBLE
        pdau RECURRING
                reception       [S] VISIBLE
        ::= id-ref-secondary


-- Secondary objects

ipms-ua OBJECT
        PORTS {
                origination     [S],
                reception       [S],
                management      [S],
                submission      [C],
                delivery        [C],
                retrieval       [C],
                administration  [C] }
        ::= id-ot-ipms-ua

ipms-ms OBJECT
        PORTS {
                submission      [S],
                retrieval       [S],
                administration  [S],
                submission      [C],
                delivery        [C],
                administration  [C] }
        ::= id-ot-ipms-ms

tlma OBJECT
        PORTS {
                origination     [S],
                reception       [S],
                management      [S],
                miscellanea     [S] }
        ::= id-ot-tlma
```

```
tlxau OBJECT
        PORTS {
                origination     [S],
                reception       [S],
                management      [S] }
        ::= id-ot-tlxau

pdau OBJECT
        PORTS {
                reception       [S]
        ::= id-ot-pdau

END  -- of IPMSFunctionalObjects
```

ANNEX G

(to Recommendation X.420)

Reference definition of abstract service

This Annex is an integral part of this Recommendation.

This Annex, a supplement to §§ 12 and 13, defines for reference purposes the IPMS abstract service. It uses the PORT and ABSTRACT-OPERATION and -ERROR macros of Recommendation X.407.

```
IPMSAbstractService { joint-iso-ccitt
        mhs-motis(6) ipms(1) modules(0) abstract-service(3) }
DEFINITIONS IMPLICIT TAGS ::=
BEGIN


-- Prologue
-- Exports everything.

IMPORTS
        -- IPMS information objects
                AutoForwardComment, Heading, IPM, NRN, RN
                ----
                FROM IPMSInformationObjects { joint-iso-ccitt
                    mhs-motis(6) ipms(1) modules(0) information-objects(2) }

        -- IPMS object identifiers
                id-pt-management, id-pt-origination, id-pt-reception
                ----
                FROM IPMSObjectsIdentifiers { joint-iso-ccitt
                    mhs-motis(6) ipms(1) modules(0) objects-identifiers(0) }

        -- MTS abstract service
                MessageDeliveryEnvelope, MessageSubmissionEnvelope,
                MessageSubmissionIdentifier, MessageSubmissionTime,
                ProbeSubmissionEnvelope, ProbeSubmissionIdentifier,
                ProbeSubmissionTime, RecipientImproperlySpecified,
                ReportDeliveryEnvelope,
                ----
                FROM MTSAbstractService { joint-iso-ccitt
                    mhs-motis(6) mts(3) modules(0) mts-abstract-service(1) }

        -- Abstract service definition conventions
                ABSTRACT-ERROR, ABSTRACT-OPERATION, PORT
                ----
                FROM AbstractServiceNotation { joint-iso-ccitt
                    mhs-motis(6) asdc(2) modules(0) notation(1) };

Time ::= UTCTime
```

```
-- Ports

origination PORT
        CONSUMER INVOKES {
                OriginateProbe,
                OriginateIPM,
                OriginateRN }
        ::= id-pt-origination

reception PORT
        SUPPLIER INVOKES {
                ReceiveReport,
                ReceiveIPM,
                ReceiveRN,
                ReceiveNRN }
        ::= id-pt-reception

management PORT
        CONSUMER INVOKES {
                ChangeAutoDiscard,
                ChangeAutoAcknowledgment,
                ChangeAutoForwarding }
        ::= id-pt-management


-- Origination abstract operations

OriginateProbe ::= ABSTRACT-OPERATION
        ARGUMENT SET {
                envelope   [0] ProbeSubmissionEnvelope,
                content    [1] IPM }
        RESULT SET {
                submission-identifier [0] ProbeSubmissionIdentifier,
                submission-time       [1] ProbeSubmissionTime }
        ERRORS {
                SubscriptionError,
                RecipientImproperlySpecified }

OriginateIPM ::= ABSTRACT-OPERATION
        ARGUMENT SET {
                envelope   [0] MessageSubmissionEnvelope,
                content    [1] IPM }
        RESULT SET {
                submission-identifier [0] MessageSubmissionIdentifier,
                submission-time       [1] MessageSubmissionTime }
        ERROR {
                SubscriptionError,
                RecipientImproperlySpecified }

OriginateRN ::= ABSTRACT-OPERATION
        ARGUMENT SET {
                envelope   [0] MessageSubmissionEnvelope,
                content    [1] RN }
        RESULT SET {
                submission-identifier [0] MessageSubmissionIdentifier,
                submission-time       [1] MessageSubmissionTime }
        ERROR {
                SubscriptionError,
                RecipientImproperlySpecified }


-- Reception abstract operations

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

```
ReceiveIPM ::= ABSTRACT-OPERATION
        ARGUMENT SET {
                envelope   [0] MessageDeliveryEnvelope,
                content    [1] IPM }
        RESULT
        ERRORS {}

ReceiveRN ::= ABSTRACT-OPERATION
        ARGUMENT SET {
                envelope   [0] MessageDeliveryEnvelope,
                content    [1] RN }
        RESULT
        ERRORS {}

ReceiveNRN ::= ABSTRACT-OPERATION
        ARGUMENT SET {
                envelope   [0] MessageDeliveryEnvelope,
                content    [1] NRN }
        RESULT
        ERRORS {}


-- Management abstract operations

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

ChangeAutoAcknowledgement ::= ABSTRACT-OPERATION
        ARGUMENT SET {
                auto-acknowledge-IPMs [0] BOOLEAN,
                auto-acknowledge-suppl-receipt-info [1]
                    SupplementaryInformation }
        RESULT
        ERRORS {
                SubscriptionError }

ChangeAutoForwarding ::= ABSTRACT-OPERATION
        ARGUMENT SET {
                auto-forward-IPMs          [0] BOOLEAN,
                auto-forward-recipients    [1] SEQUENCE OF ORName OPTIONAL,
                auto-forward-heading       [2] Heading OPTIONAL,
                auto-forward-comment       [3] AutoForwardComment OPTIONAL }
        RESULT
        ERRORS {
                SubscriptionError,
                RecipientImproperlySpecified }


-- Abstract errors

SubscriptionError ::= ABSTRACT-ERROR
        PARAMETER SET {
                problem    [0] SubscriptionProblem }

SubscriptionProblem ::= ENUMERATED {
        ipms-eos-not-subscribed          (0),
        mts-eos-not-subscribed           (1) }

END  -- of IPMSAbstractService
```

ANNEX H

(to Recommendation X.420)

Reference definition of heading extensions

This Annex is an integral part of this Recommendation.

This Annex, a supplement to Annex A, defines for reference purposes the heading extensions defined for interpersonal messaging. It uses the HEADING-EXTENSION macro of § 12.2.17.

```
IPMSHeadingExtensions { joint-iso-ccitt
    mhs-motis(6) ipms(1) modules(0) heading-extensions(6) }
DEFINITIONS IMPLICIT TAGS ::=
BEGIN


-- Prologue
-- Exports everything.

IMPORTS
    -- IPMS information objects
            HEADING-EXTENSION
            ----
            FROM IPMSInformationObjects { joint-iso-ccitt
                mhs-motis(6) ipms(1) modules(0) information-objects(2) };

    -- IPMS object identifiers
            id-hex-incomplete-copy, id-hex-languages
            ----
            FROM IPMSObjectsIdentifiers { joint-iso-ccitt
                mhs-motis(6) ipms(1) modules(0) objects-identifiers(0) };


-- Incomplete copy

Incomplete copy HEADING-EXTENSION
    ::= id-hex-incomplete-copy

IncompleteCopy::= NULL


-- Languages

languages HEADING-EXTENSION
    VALUE SET OF Language
    ::= id-hex-languages

Language ::= PrintableString (SIZE (2..2))

END -- of IPMSHeadingExtensions
```

ANNEX I

(to Recommendation X.420)

Reference definition of extended body part types

This Annex is an integral part of this Recommendation.

This Annex, a supplement to Annex B, defines for reference purposes certain extended body part types.

```
IPMSExtendedBodyPartTypes {joint-iso-ccitt
    mhs-motis(6) ipms(1) modules(0) extended-body-part-types(7)}
DEFINITIONS IMPLICIT TAGS ::=
BEGIN


-- Prologue
-- Exports everything.

IMPORTS
    -- IPMS information objects
        BilaterallyDefinedBodyPart, EncryptedData,
        EncryptedParameters, EXTENDED-BODY-PART-TYPE, G3FacsimileData,
        G3FacsimileParameters, G4Class1BodyPart, IA5TextData,
        IA5TextParameters, MessageData, MessageParameters,
        MixedModeBodyPart, NationallyDefinedBodyPart,TeletexData,
        TeletexParameters, VideotexData, VideotexParameters,
        VoiceData, VoiceParameters
        ----
        FROM IPMSInformationObjects {joint-iso-ccitt
            mhs-motis(6) ipms(1) modules(0) information-objects(2)}

    -- IPMS object identifiers
        id-ep-encrypted,
        id-ep-g3-facsimile,
        id-ep-ia5-text,
        id-ep-message,
        id-ep-teletex,
        id-ep-videotex,
        id-ep-voice,
        id-et-bilaterally-defined, id-et-encrypted, id-et-g3-facsimile,
        id-et-g4-class1, id-et-ia5-text, id-et-message,
        id-et-mixed-mode, id-et-nationally-defined, id-et-teletex,
        id-et-videotex, id-et-voice
        ----
        FROM IPMSObjectsIdentifiers {joint-iso-ccitt
            mhs-motis(6) ipms(1) modules(0) objects-identifiers(0)}


-- Extended IA5 text body part

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


-- Extended voice body part

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

```
-- Extended G3 facsimile body part

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


-- Extended G4 class 1 body part

g4-class1-body-part EXTENDED-BODY-PART-TYPE
        DATA        G4Class1BodyPart
        ::= id-et-g4-class1


-- Extended teletex body part

teletex-body-part EXTENDED-BODY-PART-TYPE
        PARAMETERS  TeletexParameters IDENTIFIED BY id-ep-teletex
        DATA        TeletexData
        ::= id-et-teletex


-- Extended videotex body part

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


-- Extended encrypted body part

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


-- Extended message body part

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


-- Extended mixed-mode body part

mixed-mode-body-part EXTENDED-BODY-PART-TYPE
        DATA MixedModeBodyPart
        ::= id-et-mixed-mode


-- Extended bilaterally defined body part

bilaterally-defined-body-part EXTENDED-BODY-PART-TYPE
        DATA BilaterallyDefinedBodyPart
        ::- id-et-bilaterally-defined


-- Extended nationally defined body part

nationally-defined-body-part EXTENDED-BODY-PART-TYPE
        DATA NationallyDefinedBodyPart
        ::- id-et-nationally-defined

END  -- of IPMSExtendedBodyPartTypes
```

ANNEX J

(to Recommendation X.420)

Reference definition of message store attributes

This Annex is an integral part of this Recommendation.

This Annex, a supplement to Annex C, defines for reference purposes the MS attributes specific to interpersonal messaging. It uses the ATTRIBUTE macro of Recommendation X.500.

```
IPMSMessageStoreAttributes { joint-iso-ccitt
    mhs-motis(6) ipms(1) modules(0) message-store-attributes(8) }
DEFINITIONS IMPLICIT TAGS ::=
BEGIN


-- Prologue
-- Exports everything.

IMPORTS
    -- IPMS heading extensions
        IncompleteCopy, Language
        ----
        FROM IPMSHeadingExtensions { joint-iso-ccitt
            mhs-motis(6) ipms(1) modules(0) heading-extensions(6) }

    -- IPMS information objects
        AcknowledgmentModeField, AuthorizingUsersSubfield,
        AutoForwardCommentField, AutoForwardedField,
        BilaterallyDefinedBodyPart, BlindCopyRecipientsSubfield, Body,
        ConversionEITsField, CopyRecipientsSubfield,
        DiscardReasonField, EncryptedBodyPart, EncryptedData,
        EncryptedParameters, ExpiryTimeField,
        ExternallyDefinedParameters, G3FacsimileBodyPart,
        G3FacsimileData, G3FacsimileParameters, G4Class1BodyPart,
        Heading, IA5TextBodyPart, IA5TextData, IA5TextParameters,
        ImportanceField, IPMPreferredRecipientField,
        IPNOriginatorField, MessageBodyPart, MessageData,
        MessageParameters, MixedModeBodyPart,
        NationallyDefinedBodyPart, NonReceiptReasonField,
        ObsoletedIPMsSubfield, ORDescriptor, OriginatorField,
        PrimaryRecipientsSubfield, ReceiptTimeField,
        RelatedIPMsSubfield, RepliedToIPMField,
        ReplyRecipientsSubfield, ReplyTimeField, ReturnedIPMField,
        SensitivityField, SubjectField, SubjectIPMField,
        SupplReceiptInfoField, TeletexBodyPart, TeletexData,
        TeletexParameters, ThisIPMField, VideotexBodyPart,
        VideotexData, VideotexParameters, VoiceBodyPart, VoiceData,
        VoiceParameters
        ----
        FROM IPMSInformationObjects { joint-iso-ccitt
            mhs-motis(6) ipms(1) modules(0) information-objects(2) }

    -- IPMS object identifiers
        id-bat-bilaterally-defined-body-parts, id-bat-body,
        id-bat-encrypted-body-parts, id-bat-encrypted-data,
        id-bat-encrypted-parameters, id-bat-extended-body-part-types,
        id-bat-g3-facsimile-body-parts, id-bat-g3-facsimile-data,
        id-bat-g3-facsimile-parameters, id-bat-g4-class1-body-parts,
        id-bat-ia5-text-body-parts, id-bat-ia5-text-data,
        id-bat-ia5-text-parameters, id-bat-message-body-parts,
        id-bat-message-data, id-bat-message-parameters,
        id-bat-mixed-mode-body-parts,
        id-bat-nationally-defined-body-parts,
```

```
            id-bat-teletex-body-parts, id-bat-teletex-data,
            id-bat-teletex-parameters, id-bat-videotex-body-parts,
            id-bat-videotex-data, id-bat-videotex-parameters,
            id-bat-voice-body-parts, id-bat-voice-data,
            id-bat-voice-parameters, id-hat-authorizing-users,
            id-hat-auto-forwarded, id-hat-blind-copy-recipients,
            id-hat-copy-recipients, id-hat-expiry-time, id-hat-heading,
            id-hat-importance, id-hat-incomplete-copy, id-hat-languages,
            id-hat-nrn-requestors, id-hat-obsoleted-IPMs,
            id-hat-originator, id-hat-primary-recipients,
            id-hat-related-IPMs, id-hat-replied-to-IPM,
            id-hat-reply-recipients, id-hat-reply-requestors,
            id-hat-reply-time id-hat-rn-requestors, id-hat-sensitivity,
            id-hat-subject, id-hat-this-ipm, id-nat-acknowledgment-mode,
            id-nat-auto-forward-comment, id-nat-conversion-eits,
            id-nat-discard-reason, id-nat-ipm-preferred-recipient,
            id-nat-ipn-originator, id-nat-non-receipt-reason,
            id-nat-receipt-time, id-nat-returned-ipm, id-nat-subject-ipm,
            id-nat-suppl-receipt-info, id-sat-ipm-entry-type,
            id-sat-ipm-synopsis
            ----
            FROM IPMSObjectIdentifiers {joint-iso-ccitt
                mhs-motis(6) ipms(1) modules(0) object-identifiers(0)}

    -- MS abstract service
            MS-EITs, SequenceNumber
            ----
            FROM MSAbstractService {joint-iso-ccitt
                mhs-motis(6) ms(4) modules(0) abstract-service(1)}

    -- MTS abstract service
            EncodedInformationTypes
            ----
            FROM MTSAbstractService {joint-iso-ccitt
                mhs-motis(6) mts(3) modules(0) mts-abstract-service(1)};

    -- Directory information framework
            ATTRIBUTE
            ----
            FROM InformationFramework {joint-iso-ccitt
                ds(5) modules(1) informationFramework(1)};

Time ::= UTCTime


-- SUMMARY ATTRIBUTES

-- IPM entry type

ipm-entry-type ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX IPMEntryType
    MATCHES FOR EQUALITY
    SINGLE VALUE
    ::= id-sat-ipm-entry-type

IPMEntryType ::= ENUMERATED{
    ipm     (0),
    rn      (1),
    nrn     (2)}


-- IPM synopsis

ipm-synopsis ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX IPMSynopsis
    SINGLE VALUE
    ::= id-sat-ipm-synopsis
```

```
IPMSynopsis ::= SEQUENCE OF BodyPartSynopsis

BodyPartSynopsis ::= CHOICE {
        messsage        [0] MessageBodyPartSynopsis,
        non-messsage    [1] NonMessageBodyPartSynopsis }

MessageBodyPartSynopsis ::= SEQUENCE {
        number          [0] SequenceNumber,
        synopsis        [1] IPMSynopsis }

NonMessageBodyPartSynopsis ::= SEQUENCE {
        type            [0] OBJECT IDENTIFIER,
        parameters      [1] ExternallyDefinedParameters,
        size            [2] INTEGER,
        processed       [3] BOOLEAN DEFAULT FALSE }


-- HEADING ATTRIBUTES

-- Heading

heading ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX Heading
        SINGLE VALUE
        ::= id-hat-heading


-- Heading analyses

rn-requestors ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ORDescriptor
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-hat-rn-requestors

nrn-requestors ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ORDescriptor
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-hat-nrn-requestors

reply-requestors ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ORDescriptor
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-hat-reply-requestors


-- Heading fields

this-ipm ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ThisIPMField
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-hat-this-ipm

originator ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX OriginatorField
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-hat-originator

replied-to-IPM ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX RepliedToIPMField
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-hat-replied-to-IPM
```

```
subject ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX SubjectField
        MATCHES FOR EQUALITY SUBSTRINGS
        SINGLE VALUE
        ::= id-hat-subject

expiry-time ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ExpiryTimeField
        MATCHES FOR EQUALITY ORDERING
        SINGLE VALUE
        ::= id-hat-expiry-time

reply-time ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ReplyTimeField
        MATCHES FOR EQUALITY ORDERING
        SINGLE VALUE
        ::= id-hat-reply-time

importance ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ImportanceField
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-hat-importance

sensitivity ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX SensitivityField
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-hat-sensitivity

auto-forwarded ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX AutoForwardedField
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-hat-auto-forward


-- Heading sub-fields

authorizing-users ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX Authorizing-UsersSubfield
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-hat-authorizing-users

primary-recipients ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX PrimaryRecipientsSubfield
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-hat-primary-recipients

copy-recipients ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX CopyRecipientsSubfield
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-hat-copy-recipients

blind-copy-recipients ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX BlindCopyRecipientsSubfield
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-hat-blind-copy-recipients

obsoleted-IPMs ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ObsoletedIPMsSubfield
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-hat-obsoleted-IPMs
```

```
related-IPMs ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX RelatedIPMsSubfield
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-hat-related-IPMs

reply-recipients ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ReplyRecipientsSubfield
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-hat-reply-recipients


-- Heading extensions

incomplete-copy ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX IncompleteCopy
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-hat-incomplete-copy

languages ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX Language
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-hat-languages


-- BODY ATTRIBUTES

-- Body

body ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX Body
        SINGLE VALUE
        ::= id-bat-body


-- Basic body parts

ia5-text-body-parts ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX IA5TextBodyParts
        MULTI VALUE
        ::= id-bat-ia5-text-body-parts

voice-body-parts ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX VoiceBodyPart
        MULTI VALUE
        ::= id-bat-voice-body-parts

g3-facsimile-body-parts ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX G3FacsimileBodyPart
        MULTI VALUE
        ::= id-bat-g3-facsimile-body-parts

g4-class1-body-parts ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX G4Class1BodyPart
        MULTI VALUE
        ::= id-bat-g4-class1-body-parts

teletex-body-parts ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX TeletexBodyPart
        MULTI VALUE
        ::= id-bat-teletex-body-parts
```

```
videotex-body-parts ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX VideotexBodyPart
        MULTI VALUE
        ::= id-bat-videotex-body-parts

encrypted-body-parts ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX EncryptedBodyPart
        MULTI VALUE
        ::= id-bat-encrypted-body-parts

message-body-parts ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX SequenceNumber
        MULTI VALUE
        ::= id-bat-message-body-parts

mixed-mode-body-parts ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MixedModeBodyPart
        MULTI VALUE
        ::= id-bat-mixed-mode-body-parts

bilaterally-defined-body-parts ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX BilaterallyDefinedBodyPart
        MULTI VALUE
        ::= id-bat-bilaterally-defined-body-parts

nationally-defined-body-parts ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX NationallyDefinedBodyPart
        MULTI VALUE
        ::= id-bat-nationally-defined-body-parts


-- Basic body part parameters component

ia5-text-parameters ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX IA5TextParameters
        MULTI VALUE
        ::= id-bat-ia5-text-parameters

voice-parameters ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX VoiceParameters
        MULTI VALUE
        ::= id-bat-voice-parameters

g3-facsimile-parameters ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX G3FacsimileParameters
        MULTI VALUE
        ::= id-bat-g3-facsimile-parameters

teletex-parameters ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX TeletexParameters
        MULTI VALUE
        ::= id-bat-teletex-parameters

videotex-parameters ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX VideotexParameters
        MULTI VALUE
        ::= id-bat-videotex-parameters

encrypted-parameters ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX EncryptedParameters
        MULTI VALUE
        ::= id-bat-encrypted-parameters

message-parameters ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MessageParameters
        MULTI VALUE
        ::= id-bat-message-parameters
```

```
-- Basic body part data components

ia5-text-data ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX IA5TextData
        MULTI VALUE
        ::= id-bat-ia5-text-data

voice-data ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX Voice-Data
        MULTI VALUE
        ::= id-bat-voice-data

g3-facsimile-data ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX G3FacsimileData
        MULTI VALUE
        ::= id-bat-g3-facsimile-data

teletex-data ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX TeletexData
        MULTI VALUE
        ::= id-bat-teletex-data

videotex-data ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX VideotexData
        MULTI VALUE
        ::= id-bat-videotex-data

encrypted-data ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX EncryptedData
        MULTI VALUE
        ::= id-bat-encrypted-data

message-data ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MessageData
        MULTI VALUE
        ::= id-bat-message-data


-- Extended body part types

extended-body-part-types ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX OBJECT IDENTIFIER
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-bat-extended-body-part-types


-- Extended body parts

-- (These attributes cannot be enumerated. See § C.3.6)

-- NOTIFICATION ATTRIBUTES

-- Common fields

subject-ipm ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX SubjectIPMField
        MATCHES FOR EQUALITY SUBSTRINGS
        SINGLE VALUE
        ::= id-nat-subject-ipm

ipn-originator ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX IPNOriginatorField
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-nat-ipn-originator

ipm-preferred-recipient ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX IPMPreferredRecipientField
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-nat-ipm-preferred-recipient
```

```
conversion-eits ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX MS-EITs
        MATCHES FOR EQUALITY
        MULTI VALUE
        ::= id-nat-conversion-eits


-- Non-receipt fields

non-receipt-reason ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX NonReceiptReasonField
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-nat-non-receipt-reason

discard-reason ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX DiscardReasonField
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-nat-discard-reason

auto-forward-comment ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX AutoForwardCommentField
        MATCHES FOR EQUALITY SUBSTRINGS
        SINGLE VALUE
        ::= id-nat-auto-forward-comment

returned-ipm ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ReturnedIPMField
        SINGLE VALUE
        ::= id-nat-returned-IPM


-- Receipt fields

receipt-time ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX ReceiptTimeField
        MATCHES FOR EQUALITY ORDERING
        SINGLE VALUE
        ::= id-nat-receipt-time

acknowledgment-mode ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX AcknowledgmentModeField
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= id-nat-acknowledgment-mode

suppl-receipt-info ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX SupplReceiptInfoField
        MATCHES FOR EQUALITY SUBSTRINGS
        SINGLE VALUE
        ::= id-nat-suppl-receipt-info

END -- of IPMSMessagesStoreAttributes
```

ANNEX K

(to Recommendation X.420)

Reference definition of upper bounds

This Annex is an integral part of this Recommendation but is not a part of the corresponding ISO International Standard.

This Annex defines for reference purposes the upper bounds of various variable-length information items whose abstract syntaxes are defined in the ASN.1 modules of prior annexes.

```
IPMSUpperBounds { joint-iso-ccitt
      mhs-motis(6) ipms(1) modules(0) upper-bounds(10) }
DEFINITIONS IMPLICIT TAGS ::=
BEGIN


-- Prologue
-- Exports everything.

IMPORTS -- nothing --;


-- Upper bounds

ub-auto-forward-comment   INTEGER ::= 256

ub-free-form-name         INTEGER ::= 64

ub-ipm-identifier-suffix  INTEGER ::= 2

ub-local-ipm-identifier   INTEGER ::= 64

ub-subject-field          INTEGER ::= 128

ub-telephone-number       INTEGER ::= 32


END  -- of IPMSUpperBounds
```

ANNEX L

(to Recommendation X.420)

Support of the interpersonal messaging service

This Annex is an integral part of this Recommendation.

The interpersonal messaging service which the IPMS provides to users is defined, in non-technical terms, in Recommendation X.400. The service comprises a number of elements of service (**IPM EOS**), each representing one aspect of the service, and each defined in one or two paragraphs of prose. The present Annex indicates in detail how the present, more technical specification realizes each IPM EOS. Equivalently, it identifies the aspects of the specification a UA, e.g., must implement for it to be said to support a particular IPM EOS.

Associated with each IPM EOS are one or more information items that may appear as IPM components. The information item associated with the sensitivity indication IPM EOS, e.g., is the sensitivity heading field. A UA, TLMA, or AU shall be said to support a particular IPM EOS upon origination or reception if, and only if, it supports upon origination or reception (see § 22.1) the information items associated with that IPM EOS.

Note 1 – The task of realizing an IPM EOS may fall, in principle, upon any of the secondary objects that result from the refinement of the IPMS. In the present context, however, it is assumed that the MTS and every MS, by virtue of their application-independence, support every IPM-EOS, and that they do so without having made special provision for any of them.

Note 2 – As described in § 14, a UA makes available to its user many of the capabilities that its MS offers. These capabilities realize the elements of the message retrieval service which is defined in Recommendation X.400. The correspondence between the elements of that service and the associated technical capabilities is given in Recommendation X.413.

Note 3 – As described in § 14, a UA makes available to its user many of the capabilities that the MTS offers. These capabilities realize the elements of the message transfer service which is defined in Recommendation X.400. The correspondence between the elements of that service and the associated technical capabilities is given in Recommendation X.411.

L.1 *Support of recipient specifier components*

Some IPM EOS are realized by means of recipient specifier components. The IPM EOS in this category are listed in the first column of Table L-1/X.420. The second and third columns identify the recipient specifier component, and the particular value of that component, that are the information items associated with each listed IPM EOS.

TABLE L-1/X.420

Support of recipient specifier components

Element of service	Recipient specifier component	Value
Non-receipt notification request	Notification-requests	nrn
Receipt notification request indication	Notification-requests	rn
Reply request indication (see also Table L-2/X.420)	Reply-requested	true

Note 1 – Recipient specifiers appear as sub-fields of the Primary Recipients, Copy Recipients, and Blind Copy Recipients heading fields.

Note 2 – Every IPM EOS except Reply Request Indication falls into exactly one category. The Reply Request Indication IPM EOS falls into two categories, as indicated in the Table.

L.2 *Support of heading fields*

Some IPM EOS are realized by means of heading fields. The IPM EOS in this category are listed in the first column of Table L-2/X.420. The second column identifies the heading fields that are the information items associated with each listed IPM EOS. In the case of the extension field, the second column also identifies, in parentheses, the relevant heading extension.

L.3 *Support of body aspects*

Some IPM EOS are realized by means of aspects of the body. The IPM EOS in this category are listed in the first column of Table L-3/X.420. The second column identifies the body aspect that is the information item associated with each listed IPM EOS.

TABLE L-2/X.420

Support of heading fields

Element of service	Heading field
Authorizing users indication	Authorizing users
Auto-forwarded indication	Auto-forwarded
Blind copy recipient indication	Blind copy recipients
Cross-referencing indication	Related IPMs
Expiry date indication	Expiry time
Importance indication	Importance
IP-message identification	This IPM
Incomplete copy indication	Extensions (incomplete copy)
Language indication	Extensions (languages)
Obsoleting indication	Obsoleted IPMs
Originator indication	Originator
Primary and copy recipients indication	Primary recipients Copy recipients
Reply request indication (see also Table L-3/X.420)	Reply time Reply recipients
Replying IP-message indication	Replied-to IPM
Sensitivity indication	Sensitivity
Subject indication	Subject

Note – Every IPM EOS except Reply Request Indication falls into exactly one category. The Reply Request Indication IMP EOS falls into two categories, as indicated in the Table.

TABLE L-3/X.420

Support of body aspects

Element of service	Body aspect
Body part encryption indication	Encrypted body part
Forwarded IP-message indication	Message body part
Multi-part body	Body with two or more parts
Typed body	Body (itself)

Note – Support of the Typed Body IPM EOS is intrinsic to any implementation of any secondary object.

ANNEX M

Differences between CCITT Recommendation and ISO Standard

This Annex is not a part of this Recommendation.

This Annex lists all but the purely stylistic differences between this Recommendation and the corresponding ISO International Standard.

The following are the differences that exist:

a) The ISO International Standard corresponding to this Recommendation defines a general text body part, while this Recommendation does not.

b) The upper bounds of Annex K are an integral part of this Recommendation but are not a part of the corresponding ISO International Standard.

c) The CCITT text on subject recipient specifier in § 8 states that it may contain either "an O/R name of the preferred recipient" or "an O/R name appearing in the message's DL expansion history". The ISO Standard has excluded this second possibility.

ANNEX N

Summary of changes to 1984 specification

This Annex is not a part of this Recommendation.

Editorially, this Recommendation differs substantially from Recommendation X.420 (1984). Technically, however, the differences are few. The present Annex lists the technical changes. It is intended as an aid to an implementor of Recommendation X.420 (1984), enabling him to see at a glance how his implementation might be affected by the 1988 specification.

The following, and only the following, substantive changes relevant to interworking between 1984 and 1988 UAs, MSs, TLMAs, and AUs are embodied in the present specification. All but the first are changes to the format of the information objects now defined in the ASN.1 module, IPMSInformationObjects:

a) The content type assigned to P2 has changed. Formerly identified by the integer 2, P2 now is identified by either the integer 2 or 22, depending upon the functionality employed in a particular instance of communication by means of the MTS (see § 20.2.).

b) The omission of the user members of IPMIdentifier is now denigrated.

c) The extensions member has been added to heading. Its grade is optional.

d) The telex and simple formattable document body part types have been abandoned. (The former had been identified but not defined.)

e) The syntax member has been added to VideotexParameters. Its grade is optional.

f) The presence of the delivery-time member of the MessageParameters in the absence of its delivery-envelope member, or vice-versa, is now denigrated.

g) The bilaterally-defined and externally-defined alternatives have been added to BodyPart.

h) The following protocol elements, defined in Recommendation X.411 and incorporated in protocol elements of this Recommendation by reference, have changed:
   i) ORName
   ii) ORAddress
   iii) MessageDeliveryEnvelope
   iv) EncodedInformationTypes
   v) SupplementaryInformation.

i) Specifying a value of zero length of any of the following data types is now denigrated:
   i) LocalIPMIdentifier
   ii) FreeFormName
   iii) TelephoneNumber
   iv) SubjectField
   v) AutoForwardComment.

j) Upper bounds have been imposed upon certain variable-length protocol elements.

Note – The upper bounds imposed are those found in § 4.3 of Version 6 of the *X.400-series Implementor's Guide.*

ISBN 92-61-03721-6