ECMA

EUROPEAN COMPUTER MANUFACTURERS ASSOCIATION

STANDARD ECMA-122

MIDA

MAILBOX SERVICE DESCRIPTION AND MAILBOX ACCESS PROTOCOL SPECIFICATION

July 1987

Free copies of this document are available from ECMA,
European Computer Manufacturers Association
114 Rue du Rhône – 1204 Geneva (Switzerland)

ECMA

EUROPEAN COMPUTER MANUFACTURERS ASSOCIATION

STANDARD ECMA-122

MIDA

MAILBOX SERVICE DESCRIPTION AND MAILBOX ACCESS PROTOCOL SPECIFICATION

July 1987

BRIEF HISTORY

This ECMA Standard is one of a set of standards for Open Systems Interconnection. Open Systems Interconnection standards are intended to facilitate homogeneous interconnection between heterogeneous information processing systems. The Standard is within the framework for the coordination of standards for Open Systems Interconnection which is defined by ISO 7498.

This ECMA Standard is based on the practical experience of ECMA member companies world-wide, and on the results of their active participation in the current work of ISO, CCITT and national standard bodies in Europe and the USA. It represents a pragmatic and widely based consensus.

A particular emphasis of this Standard is to specify the homogeneous externally visible and verifiable characteristics needed for interconnection compatibility, while avoiding unnecessary constraints upon and changes to the heterogeneous internal design and implementation of the information processing systems to be interconnected.

In the interest of a rapid and effective standardisation, the Standard is oriented towards urgent and well understood needs. It is intended to be capable of modular extension to cover future developments in technology and needs.

Adopted as an ECMA Standard by the General Assembly of ECMA on June 25, 1987.

TABLE OF CONTENTS

1. SCOPE

This ECMA Standard for a MIDA Mailbox Service Description and Mailbox Access Protocol Specification:

- Defines, based on the Client-Server Model as defined in ECMA TR/42 Framework for Distributed Office Applications, operations to access the services provided by the Mailbox Server;
- Specifies the Mailbox Access Protocol to support the above services and its mapping onto the underlying services;
- Specifies the requirements for Conformance with the Mailbox Access Protocol.

The Mailbox Access Protocol defined in this Standard enables a User to use the Mailbox Server, but does not include service administration aspects. The administrative functions can be provided by using other Applications such as Directory.

The potential use by the Mailbox Servers of other standardized Applications is described in Appendix C to this Standard.

This Standard is for a Distributed Office Application located in the Application Layer of the Basic Reference Model of Open Systems Interconnection (see ISO 7498).

2. FIELD OF APPLICATION

The ECMA MIDA Standard, ECMA-93 second edition, describes a general message handling service.

This Standard describes extensions to the MIDA model as defined in ECMA-93 with the following objectives:

- To introduce a distribution of the User Agent functionality, as defined by the User Agent (UA) concept in ECMA-93, into one part which offers storage, the Mailbox Server, and one part which contains the more user-oriented functions of a UA, the Mailbox Client.
- The Mailbox Server offers storage of incoming messages and notifications and allows the Mailbox Client to fetch those messages and notifications at the Mailbox Client's convenience.
- The Mailbox Server also offers the facility to accept messages from the Mailbox Client for submission to the Message Transfer Service. There is no facility for the Mailbox Client to fetch back messages which have been dispatched to the Mailbox Server.
- The Mailbox Server may also offer facilities to keep a log over all incoming and outgoing messages (and notifications). The primary purpose of the log is to store certain key information as proof of what has been delivered into the Mailbox, or has been dispatched by a Mailbox Client over a particular period of time.
- The Mailbox Server may also offer facilities to correlate notifications and replies with the original message accepted for submission.
- The Mailbox Server may also offer the facility to accept registration information destined for the associated Message Transfer Agent from the Mailbox Client.
- The Mailbox Server makes the Message Transfer Services available to the Mailbox Client.
- If the Mailbox Server is supporting the "General Subset", the Mailbox Server is limited to handling of complete message contents only, of arbitrary content type.
- If the Mailbox Server is supporting the "IPM Subset", it also makes the IPM Service available to the Mailbox Client. It may also offer facilities to forward, reply to, or acknowledge receipt of, incoming messages.

- The Mailbox Client acts on behalf of a User, identified by name and verified by credentials. Several different Users may access one Mailbox. The Mailbox Server may offer facilities to record which Users perform which actions on the incoming messages and notifications.

3. CONFORMANCE

3.1 General

This clause defines the conformance requirements for the Mailbox Access Protocol specified in clause 10 of this Standard.

A system providing Services and Protocol as described in this Standard is called a MIDA Mailbox System. A MIDA Mailbox System can operate either as a Mailbox Server or a Mailbox Client or both.

The requirements to support elements of the Protocol as specified in clause 10 depend on the requirements for support of Attributes as defined in clause 8 and the requirements for support of Services as defined in clause 9. The requirements for support of Attributes and Services is expressed in support being either Mandatory or Optional as defined below in clauses 3.6.and 3.7.

Only the externally visible and externally testable criteria are defined.

3.2 Equipment

The conformance requirement is for equipment which consists of hardware and/or software and has the purpose of conforming with this Standard. The equipment may also have other purposes.

3.3 Peer Equipment

Any execution of the Protocol necessarily involves a peer equipment with which the subject equipment communicates. For purposes of verifying conformance, it is assumed that this other peer equipment:

- is operating in conformance with this Standard;
- may be capable of controlled deviation, in that it may be the source of deliberate protocol errors for the purpose of testing.

3.4 Protocol Subsets

Two Protocol subsets are defined within this Standard.

- The first subset is called the "General Subset" and contains the protocol elements necessary to support the General Mailbox Services (see clause 9.2).
- The second subset is called the "IPM Subset" and contains the General Subset together with the Protocol elements necessary to support the IPM-specific Service extensions (see clause 9.3).

3.5 Additional Message Interchange Protocols

In addition to the Protocol defined in this Standard, the equipment may also implement other message interchange protocols.

Such additional provisions are themselves not in conformance with this Standard, but do not prejudice conformance with this Standard provided that they are separate and do not prevent use of the Protocol defined within this Standard.

3.6 Service Classification

The Mailbox Services are classified as:

- Mandatory;
- Optional.

The interpretation of these terms is:

Mandatory

A Service whose Protocol support is mandatory in any system claiming conformance to this Standard. The associated Protocol elements and encoding as defined herein must be supported (but note: a Mailbox Client may choose not to invoke all the Service features offered as Mandatory). The actions that must be taken to support a Service may depend on the role of the equipment as either being a Mailbox Server or a Mailbox Client. Note that the set of Mandatory Services depends on the Subset to which conformance is claimed.

Optional

A Service whose Protocol support is not mandatory. If the Service is supported, however, it must obey the Protocol elements and encoding as defined in this Standard. A Mailbox Client may choose not to invoke some or all Service features offered as optional.

3.7 Attribute Classification

The Mailbox Attributes are classified as:

- Mandatory;

- Optional.

The interpretation of these terms is:

Mandatory

An Attribute whose Protocol support is mandatory in any system claiming conformance to this Standard. The associated Protocol elements and encoding as defined herein must be supported. The actions that must be taken to support an Attribute may depend on the role of the equipment as either being a Mailbox Server or a Mailbox Client. Note that the set of Mandatory Attributes depends on the Subset to which conformance is claimed.

Optional

An Attribute whose Protocol support is not mandatory. If the Attribute is supported, however, it must obey the Protocol elements and encoding as defined in this Standard. A Mailbox Client may choose not to subscribe to some or all Attribute features offered as optional.

3.8 Requirements

As a minimum, each equipment conforming to this Standard must conform to the syntax and semantics of the Mandatory Mailbox Access Protocol elements of the General Subset in either the role of a Mailbox Client or a Mailbox Server.

A MIDA Mailbox System shall use the Remote Operation Service (ROS) as defined in ECMA TR/31.

3.8.1 Conformance to the Mailbox Access Protocol

The subject equipment:

- shall accept all correct APDUs containing messages, for the Subset to which it claims to support, received from peer equipment;

- shall only generate correct APDUs containing messages to peer equipment.

- shall map such messages onto lower layers as described in clause 11 of this Standard.

The conformance statement must specify which role (Client or Server) the subject equipment supports and which Subsets it can support in that role.

The conformance statement must specify which (if any) of the Optional Services for the Subset it can support, which for Reception and which for Origination.

The conformance statement must also specify which (if any) of the Optional Attributes for the Subset it can support.

4. REFERENCES

ECMA-93	Distributed Application for Message Interchange (MIDA) (Second Edition)
ECMA-101	Office Document Architecture (ODA)
ECMA TR/31	Remote Operations: Concepts, Notation and Connection-oriented Mappings
ECMA TR/32	OSI Directory Access Service and Protocol
ECMA TR/42	Framework for Distributed Office Applications
ISO 7498	Data Processing-Open Systems Interconnection-Basic Reference Model
ISO DIS 8824.2	Specification of Abstract Syntax Notation One (ASN.1). July 1986
ISO DIS 8825.2	Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1). July 1986
CCITT Rec X.411	Message Handling Systems: Message Transfer Layer (1985)

5. DEFINITIONS AND ACRONYMS

5.1 Definitions

5.1.1 General Terminology for Message Handling

The following terms are used as defined in ECMA-93 (MIDA):

Auto-forwarded Indicator
Auto-forwarding
Body
Body Part
Body Part Type
Content
Deferred Delivery
Delivery
Delivery Envelope
Delivery Notification
Delivery Time
Encoded Information Type
Entity
Envelope
Expiry Date
Heading
Interpersonal Messaging (IPM)
IP-message
IPM Content
IPM Service
IPM-status-report
Message
Message Transfer Agent
Message Transfer Sublayer
Non-delivery Notification
Non-receipt
Non-receipt Notification
O/R Name
Originator
Probe Request
Receipt
Receipt Notification

Recipient
Reply/Replies
Return
Returned Content
Returned IP-message Content
Service Message
Submit Event Id
User
User Agent
User Agent Sublayer
User Message

5.1.2 Terminology for Remote Operations

The following term is used as defined in ECMA TR/31 Remote Operations: Concepts, Notation and Connection-oriented Mappings:

Operation Macro

5.1.3 Terminology for Distributed Office Applications

The following terms are used as defined in ECMA TR/42 Framework for Distributed Office Applications:

Access Control List
Access Protocol
Attribute
Client
Client-Server Model
Server
Third Party Transfer

5.1.4 Terminology Specific to this Standard

5.1.4.1 Action

An activity related to a message in the Message Store. Actions can be either user-initiated (such as fetch or delete) or automatic, performed by the MBS (such as auto-forward).

5.1.4.2 Action History

An Attribute in the MS and/or Inlog Entries containing a list of the Actions performed on a specific MS Message.

5.1.4.3 Auto-correlation

A function performed by the MBS. Auto-correlation is requested by the MBC on a per message basis for submitted messages. Returns are matched and recorded against information about the submitted message.

5.1.4.4 Auto-correlation Entry

The auto-correlation information associated with one submitted message, for which auto-correlation was requested.

5.1.4.5 Auto-correlation Log

The object containing all the Auto-correlation Entries.

5.1.4.6 Auto-Non-receipt

An automatic Action performed by the MBS and resulting in a Non-receipt Notification being sent back to the originating UA.

5.1.4.7 Auto-Receipt

An automatic Action performed by the MBS and resulting in a Receipt Notification being sent back to the originating UA.

5.1.4.8 Comment

A fixed text to be included as part of a Non-receipt Notification which was sent as a result of auto-forwarding.

5.1.4.9 Deleted (MS-Status)

A value of the MS-Status Attribute in an Inlog. In this case, the complete message which was delivered into the Message Store has been deleted and only the Inlog Entry information is still available.

5.1.4.10 Dispatch

The process by which messages are passed from an MBC to the MBS for subsequent submission to an MTA. A dispatch is not considered to be successful until the corresponding submission is successful.

5.1.4.11 Dispatch Sequence Number

An Attribute allocated by the MBS to each dispatched message or probe. Dispatch Sequence Numbers ascend in chronological order.

5.1.4.12 Entry

An Entry consists of a set of Attributes related to a particular message which has been handled by the MBS. Each of the Mailbox objects have entries, e.g. MS Entry, Inlog Entry.

5.1.4.13 IP-message Synopsis

An Attribute describing the internal structure of an IP-message. It shows for each body part the type, the length and whether the body part has been explicitly fetched by a User or not.

5.1.4.14 Inlog

The object containing all the Inlog Entries. The intent of the Inlog is to keep, on a long term basis, certain information about each message delivered into the Message Store, even after the message has been deleted.

5.1.4.15 Listed (MS Status)

The MBC has performed some action or actions on the MS Message, but not sufficient to deem the message to be processed.

5.1.4.16 Mailbox (Object)

This is the main object found in a Mailbox Server. An association between an MBC and an MBS is limited to one Mailbox object. From the Message Transfer Service point of view, a Mailbox is a User Agent and hence the handing over of a message from an MTA to an MBS constitutes a delivery.

5.1.4.17 Mailbox Application

The application defined in this Standard.

5.1.4.18 Mailbox Access Protocol (P7)

The access protocol specified in this Standard.

5.1.4.19 Mailbox Client (MBC)

The entity in a Mailbox communication which is colocated with the User.

5.1.4.20 Mailbox Server (MBS)

The entity in a Mailbox communication which contains the Mailbox object.

5.1.4.21 Message Store

An object within the Mailbox containing all incoming messages awaiting either on automatic action or an action triggered by a Mailbox user via the MBC.

5.1.4.22 MS Content

The part of an MS Message that comprises the content of the message.

5.1.4.23 MS Envelope

The part of an MS Message that comprises delivery information or notify information.

5.1.4.24 MS Message

A message in the Message Store. It consists of an MS Entry, an MS Envelope and an MS Content.

5.1.4.25 MS Sequence Number

An Attribute allocated by the MBS to each message delivered into the Message Store. MS Sequence Numbers ascend in chronological order.

5.1.4.26 MS Status

An Attribute allocated by the MBS to each message delivered into the Message Store. The value of MS Status gives an indication about the Actions performed by Users (via the MBC) or automatically by the MBS. The values are : New, Listed, Processed and (for Inlog only) Deleted.

5.1.4.27 New (MS Status)

The message has not yet been noted by a User or automatically processed by the MBS.

5.1.4.28 Outlog

The object containing all the Outlog Entries. The intent of the Outlog is to keep, on a long term basis, certain information about each message or probe dispatched.

5.1.4.29 Owner

In the content of the Mailbox application, the primary User of a Mailbox with full access rights and who may permit subsidiary Users to access the Mailbox by entering them in the corresponding Access Control List. The latter is outside the scope of the Standard.

5.1.4.30 Processed (MS Status)

Means that the message was either completely fetched by a User or was automatically processed by the MBS. For details see clause 7.2.3.

5.1.4.31 Return Status

Information about received or not received returns for a specific value of the Auto-correlation element.

5.1.4.32 Subscription

Subscription refers to a long term agreement between parties on services made available by one party to another party.

For the purpose of this standard it refers to services made available by the MBS to the MBC. In particular the MBS is assumed to register the following subscription information for each Mailbox :

- the optional services that users of the Mailbox may perform. As far as subscription is concerned this is the same set for all users of the Mailbox. Mandatory services may always be performed.
- the optional attributes that the Mailbox maintains in each of the following Mailbox Objects; Message Store, Inlog and Outlog. This is the same for all users of the Mailbox. Mandatory attributes are always maintained in the relevant MB Objects.

The loading and maintenance of subscription information is a local matter and outside the scope of this Standard.

5.1.4.33 Subsidiary Users

Users of a Mailbox that are not Owners.

5.1.4.34 Submission and Delivery Protocol (P3)

A protocol, defined in the CCITT Rec. X.411, which governs the communication between a Submission and Delivery Entity and a Message Transfer Agent Entity.

5.2 Acronyms

For the purpose of this Standard the following acronyms apply:

AC	Auto-correlation
EIT	Encoded Information Type
IP	Interpersonal
IPM	Interpersonal Messaging
MB	Mailbox
MBC	Mailbox Client
MBS	Mailbox Server
MIDA	Distributed Application for Message Interchange
MS	Message Store
MTA	Message Transfer Agent
MTS	Message Transfer System (not Service)
MTSL	Message Transfer Sublayer
O/R	Originator/Recipient
OSI	Open Systems Interconnection
P1	Message Transfer Protocol
P2	Interpersonal Messaging Protocol
P3	Submission and Delivery Protocol
P7	Mailbox Access Protocol
Pc	Family of Content Protocols
PSAP	Presentation Service Access Point
ROS	Remote Operation Service
RTS	Reliable Transfer Server
UA	User Agent
UASL	User Agent Sublayer

6. OVERVIEW

6.1 General Objectives

This Mailbox Standard allows, in conjunction with other ECMA Standards, a Message Handling System to be configured such that the User Agent function may be distributed between a Mailbox Server and a Mailbox Client.

This Standard makes a full use of the OSI reference model and of its Standards:

- The Mailbox Access mechanism may be used between open systems which implement Mailbox Access Protocols.
- The need for new protocols has been minimized by re-using existing protocols wherever possible.
- A Mailbox Server and a Mailbox Client may be interconnected without any assumption on the interconnection service which could be available; leased lines, PDNs, or LANs, for example, may support this mechanism.

The areas addressed by this Mailbox Standard are :

- Description of a model for the MIDA Mailbox Server and the objects stored in a Mailbox,
- Description of the Attributes that can occur in a Mailbox,
- Description of the Mailbox Services,
- Specification of the Mailbox Access Protocol,
- Description of the usage of Lower Layer Services.

6.2 Mailbox Access Principles

MIDA systems offer asynchronous communication between an undetermined number of users and provide store and forward capabilities.

The MIDA Standard, ECMA-93, defines the services and specifies the protocols associated with this store and forward communication.

The Mailbox Access Protocol allows the User Agent described in Standard ECMA-93 to be distributed between a Mailbox Client (MBC) and a Mailbox Server (MBS).

The Mailbox Access Protocol allows the Mailbox Client to make full use of the Message Transfer and, optionally, the Interpersonal Messaging Services with the added value of a delivered message store and a retrieval facility. It also offers logging facilities.

Some additional Message Transfer Services are defined for the situation where the UA and the MTA are not co-located. Since they would only be of local significance in a co-located UA/MTA system, they are not defined in ECMA-93. The Mailbox Access Protocol conveys these services to the Mailbox Client.

6.3 Client-Server Model

The Mailbox Access is based on a Client-Server model, as defined in ECMA TR/42 Framework for Distributed Office Applications (see Fig. 1).

The Mailbox System contains a number of Mailbox Servers each responsible for a set of Mailboxes.

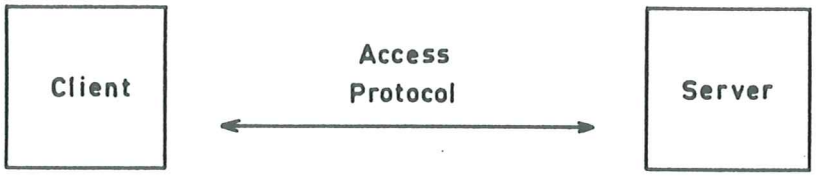

Figure 1 : The Client - Server Model.

7. MIDA MAILBOX SERVER MODEL

This MIDA Mailbox Standard describes the Mailbox Service, and specifies the protocol between a Mailbox Client and a Mailbox Server when the Mailbox Client is located remotely from the Mailbox Server.

From the Message Transfer System point of view, a Mailbox Server is a set of User Agents, each of which is termed a Mailbox and represents a special case of the general User Agent concept of ECMA-93, in that as well as fulfilling the User Agent Sublayer Services of ECMA-93, it embodies specific service extensions as described in this Standard. These additional services do not need standardization when the Mailbox Client is local to the system providing the Mailbox Server, but have been standardized to allow the definition of a Mailbox Access Protocol (P7) for use when the Mailbox Client accesses the Mailbox Server from a remote system.

The MIDA Mailbox Server model allows the concept of a multi-user Mailbox (e.g. a departmental Mailbox, manager/secretary Mailbox) which, although being a unique and singular MIDA UA, allows multiple (and concurrent) access by users with appropriate access rights.

7.1 Mailbox Access Overview

7.1.1 Mailbox Server Access Layered Model

Figure 2 illustrates the layered model of the Mailbox Service and its relationships to the MIDA model from ECMA-93.

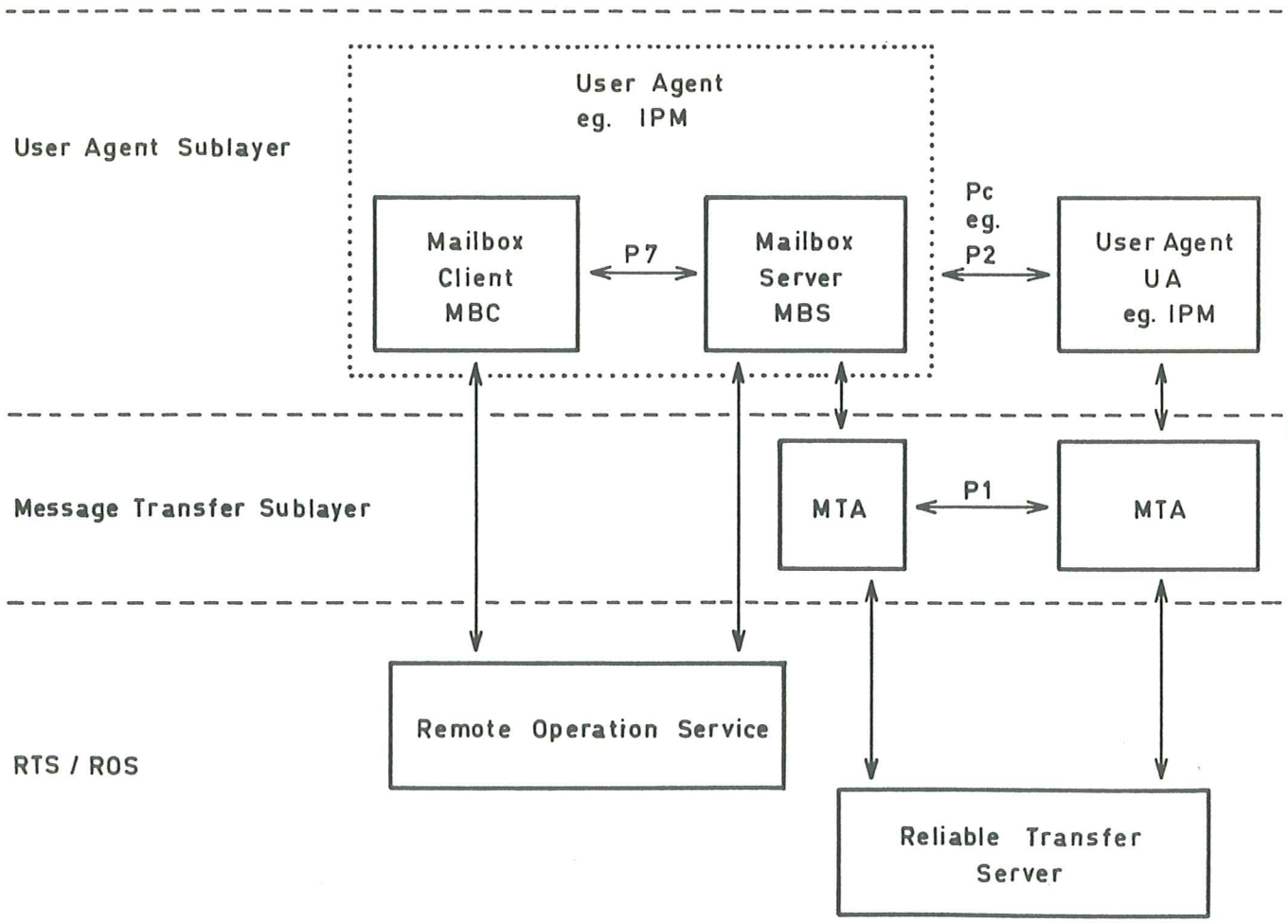

Figure 2 : Layered Model of Mailbox Access

Note 1

Although not described in the model in Figure 2, the left-hand MTA and the MBS can be located in different systems, communicating by means of the CCITT P3 protocol, specified in CCITT Rec. X.411.

7.1.2 Structure of the Mailbox Service

The abstract definition of the Mailbox Server applies to the concepts of Remote Operations (ECMA TR/31).

Mailbox is an object type defined by its external visible behaviour, specified by its Remote Operations.

A specific object of type Mailbox is a single Mailbox. A collection of ingle Mailboxes with their associated management function is referred to as a Mailbox Server.

A Mailbox Server contains:

- a set of Mailboxes;

- a register of Mailbox Users.

From the Message Transfer System point of view, a Mailbox Server is a set of User Agents.

A single Mailbox corresponds to a single User Agent and is:

- identified by an O/R Name, unique within the Message Handling System;

- assigned to and accessible by a single User, referred to as the "Owner" of that Mailbox;

- accessible to a set of Subsidiary Users.

A single User is:

- identified by a User Name (unique at least within the scope of the Mailbox Server);

- authenticated by means of User credentials (e.g. password);

- represented by a Mailbox Client.

The Owner of a Mailbox may permit Subsidiary Users to access this Mailbox.

The object binding occurs between a single User and a single Mailbox. Therefore the User Name, the User's credentials and the O/R Name of the Mailbox have to be specified in the BIND-operation.

The operations a User may invoke may depend on whether the User is the Owner of the Mailbox or a Subsidiary User.

7.1.3 Mailbox Server Functions

A Mailbox Server is a functional component of a Message Handling System which provides its Clients with certain services over and above those provided by the Message Transfer System of the MIDA model. Some of these Message Transfer services are made available transparently by the Mailbox Server.

The main feature of the Mailbox Server is to provide storage of incoming messages, and notifications for an indefinite time. This storage is termed the Message Store (MS). Facilities are provided to allow the Client system to "fetch" messages out of the MS, to "list" the contents of the MS, to "delete" items from the MS, and to "forward", to "reply" to, and to acknowledge "receipt" of, messages stored in the MS.

The facility to "submit" messages to the MTS is made available to the MBC. This may be made available transparently, or, for IPM, the MBS may provide added functionality to assist the MBC. The request to submit a message may be accompanied by a request to correlate the notifications of (non) delivery and, for IPM, the notifications of (non-)receipt and the replies.

Note 2

Outgoing messages are not stored in the Mailbox. However, certain parts of the message may be stored in the optional Outlog (see clause 7.2.5).

A facility is provided to "register" with the MBS requests to automatically perform certain IPM actions, such as Receipt Notification, Non-receipt Notification, Auto-obsoleting, Auto-forwarding and Auto-expiry.

Optional "logging" facilities may be provided by the MBS. This provides a long-term record of all messages submitted and delivered between the MBS and the MTS, together with status information. Facilities for accessing the logs are then also provided. This Standard allows flexibility with respect to the information stored and accessed in logs.

A single Mailbox can serve a number of registered Users, either locally or remotely, each represented by an MBC. The MBS may provide services to record the actions performed on messages in the MS, whether initiated by a User or performed automatically by the MBS.

To distinguish between the passing of messages between the MBC and the MBS, and between the MBS and the MTA, the following terminology is used :

Dispatch = the interaction by which messages are passed from an MBC into the MBS for transfer to one or more Recipients.

Fetch = the interaction by which messages are extracted from the MBS and passed to the MBC at its request.

Submit = the interaction by which information is passed from the UASL to the MTSL for transfer;

Deliver = the interaction by which a message is passed from the MTSL to the UASL, either to a UA or to an MBS.

7.1.4 IP-message Structure

A Mailbox Server may support services which are specific for IP-messages and IPM-status-reports and which use the internal structure of such messages.

The general structure of an IP-Message is defined in ECMA-93. It can be regarded as a tree structure, as illustrated by Figure 3.

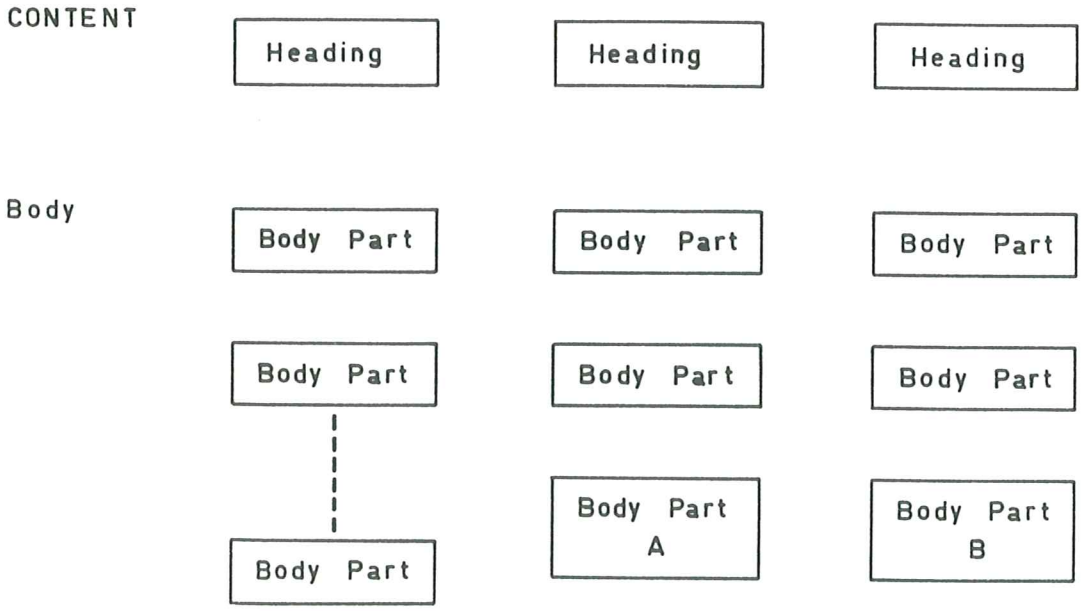

Figure 3 : Structure of an IP-message

For the purpose of interworking between MBC and MBS it can be necessary to refer to a specific body part in an IP-message or to all body parts or a sequence of some body parts within a containing body part.

A body part is identified in its constituent structure by an integer representing its relative position (heading being 0, first body part being 1, etc.) and by prefixing that integer by the integer representing the relative position of the containing structure in turn in its containing structure until the root.

So in Figure 3 Body part A is identified by 2.3. Body part B is identified by 2.2.3.

7.1.5 Mailbox Server Functionality

The following items play a role in the functionality offered to an MBC by an MBS.

- The functionality supported (i.e. implemented) by the MBS.
- The functionality subscribed to for a particular Mailbox, which is a subset of the functionality supported by the MBS. This subscription determines the information that is maintained for a particular Mailbox.
- The access rights of a particular User (represented by an MBC) when accessing a particular Mailbox. These access rights determine amongst others the information in the Mailbox available to the MBC.

During the binding of an MBC and an MBS there is no full negotiation on the optional features that are supported by both partners. Therefore it may occur that one of the partners refers to a feature not supported by the other partner.

Provided that the features referred to are within the capabilities of interworking that have been agreed for the duration of the binding, such references should not disrupt the normal interworking, i.e. where possible an abnormal condition should be returned, otherwise the not supported information should be ignored.

7.2 Mailbox Objects

7.2.1 Overview

A Mailbox may contain a number of objects, known as Mailbox Objects, for storing messages and for logging information about messages which have been handled by the Mailbox.

The following Mailbox Objects are currently defined:

- Message Store, always present in a Mailbox;
- Inlog, an optional Mailbox Object;
- Outlog, an optional Mailbox Object;
- Auto-correlation Log, an optional Mailbox Object.

The optional Mailbox Objects can be present independently of each other.

Each Mailbox Object consists of a set of Entries.

Incoming messages (deliveries or notifications) are identified by an MS Sequence Number allocated by the MBS. These ascend in chronological order, without cycling, and are never re-used. For example, the MBS may choose to allocate them by using the time to a sufficient granularity to ensure uniqueness. Entries in the Message Store and the Inlog are identified by their MS Sequence Number.

Incoming messages also have a Delivery Time. This is the date and time of either delivery or of notification, as appropriate. But note that two or more consecutive messages may have the same Delivery Time.

Outgoing messages (dispatches) are identified by a Dispatch Sequence Number allocated by the MBS. These also ascend in chronological order, without cycling, and are never re-used. Entries in the Outlog and the Auto-correlation Log are identified by their Dispatch Sequence Number.

Outgoing messages also have a Submission Time. This is the date and time of either submission or of a probe, as appropriate.

The Message Store (MS) is used to store all incoming messages awaiting either an automatic Action or an Action triggered by the MBC. Each entry in the MS contains information relating to one incoming message. In addition the MS may contain a history of the Actions performed on messages in the MS, whether initiated by the MBC or performed automatically by the MBS. Note that there is no storing of messages that the MBS has submitted to the MTSL.

The Inlog contains, on a long term basis, a summary record of all messages delivered into the MS. These summary records are compiled automatically by the MBS, and may be examined by the MBC. Each entry in the Inlog contains information relating to one incoming message. In addition, the Inlog may contain a history of the Actions performed on messages in the MS, whether initiated by the MBC or performed automatically by the MBS. This object is only relevant in cases where the MBS has implemented the Inlog Services and where subscription to the facility has been requested. Cross-referencing between the Inlog and the MS can be done using the MS Sequence Number, and from the Inlog to the Outlog using the Dispatch Sequence Number of a related dispatch.

The Outlog contains, on a long term basis, a summary record of all messages dispatched by the MBC. These summary records are compiled automatically by the MBS, and may be examined by the MBC. Each entry in the Outlog contains information relating to one dispatched message. The Outlog information is a subset of the information provided with the dispatched messsage. This object is only relevant in cases where the MBS has implemented the Outlog Services and where subscription to the facility has been requested. Cross-referencing between the Outlog and the Auto-correlation Log can be done using the Dispatch Sequence Number.

The Auto-correlation Log contains detailed information about messages dispatched by the MBC, and any returns (Notifications and replies) relating to these messages. Each entry in the Auto-correlation Log contains information relating to one dispatched message. This object is only relevant in cases where the MBS has implemented the Auto-correlation Services and where subscription to the facility has been requested. The auto-correlation logging is requested on a per-message basis. Cross-referencing between the Auto-correlation Log and the Mail Outlog can be done using the Dispatch Sequence Number, and from the Auto-correlation Log to the Message Store and to the Inlog using the MS Sequence Number of the Auto-correlated message.

7.2.2 Use of Attribute Concept

When the MBC obtains information from the Mailbox Objects, some of the information is returned using the "Attribute" mechanism described in TR/42 on Framework for Distributed Office Applications.

The following clauses, which describe the Mailbox Objects, identify the Attributes and the other information that can be made available to the MBC via the various Objects.

A Mailbox Attribute is identified by its type. The information in a Mailbox Object entry for a particular Attribute consists of a type and value pair, or, in the case of an Attribute which is defined to be "multi-valued", a type and a set of zero or more values.

Tables 1 to 4 (in clause 8) define the various Mailbox Attributes and whether they are always present, conditionally present, or always absent in a particular type of Mailbox Object entry.

The tables also define for each Attribute:

- whether or not support by the MBS and MBC is mandatory or optional (see clause 3).
- whether or not the Attribute may be used in a filter (see clause 7.2.8)
- whether or not the Attribute may be used in a synopsis request (see clause 9)
- whether the Attribute is single valued or multi-valued.

Subscription to optional attributes is per optional Attribute per Mailbox Object.

7.2.3 Message Store

The information stored for an incoming message in the Message Store is called an MS Message, or simply a "message" where the context admits of no confusion. The MS Message may either be a delivered message or a (delivery or non-delivery) notification.

An MS Message consists of three elements, each of which can be individually fetched :

- The MS Entry consists of Attributes constructed mainly from the parameter information from the DELIVER or NOTIFY service primitives as defined in ECMA-93, and, for the IPM-specific Mailbox Attributes, from the IP-message heading or the IPM-status-report. The Attributes which may be present are given in Tables 1 and 3. They include the MS Sequence Number, the Delivery Time, and an MS Status. The MS Status may take the following values:

  New — The MS Message has neither been "listed" by an MBC nor has it been automatically processed by the MBS.

  Listed — An MBC has performed some action or actions on the MS Message, but not sufficient to deem the message to be processed.

  Processed — Either an MBC has completely "fetched" the MS Message, or the MBS has performed some automatic Action (see clause 7.2.7) on it. (Note that some automatic Actions result in the MS Message being deleted). In the case of a delivered message, "completely fetched" refers to the MS Content, unless the delivered message is an IPM-status-report in which case retrieval of the returned contents (if any) is not necessary to adopt this state. In the case of a (Non-)Delivery Notification it refers to the retrieval of the MS Envelope.

- The MS Envelope consists of information constructed from the parameter information from the DELIVER and NOTIFY service primitives as defined in ECMA-93. The information is made available as one structured data element.
- The MS Content consists of the information contained in the message Content in the case of a Message or of the Delivery Report Content in case of a (Non-)Delivery Notification.

An MS Message is deleted from the MS as a whole, independently of the deletion of any other MS Message.

7.2.4 Inlog

An entry in the Inlog is called an Inlog Entry.

The Attributes that may be present in an Inlog Entry are the same as those that may be present in an MS Entry; they are listed in Tables 1 and 3.

The MBS must, for optional Attributes, offer the same support for both Inlog Entry and MS Entry. On the other hand, for a particular Mailbox the subscription may be different for the MS Entry and the Inlog Entry, reflecting the long-term nature of the Inlog Entry compared with the comparatively transient existence of an MS Message.

The MS Status Attribute may take one additional value in an Inlog Entry:

Deleted — The corresponding MS Message has been deleted from the MS, either explicitly by the MBC, or automatically by the MBS.

An Inlog Entry is deleted from the Inlog as a whole; this deletion is only possible if all older Inlog Entries have been deleted or are deleted.

7.2.5 Outlog

An entry in the Outlog is called an Outlog Entry. It consists of Attributes constructed mainly from the parameter information from the SUBMIT or PROBE service primitives as defined in ECMA-93, and, for the IPM-specific Mailbox Attributes, from the IP-message heading or the IPM-status-report. The Attributes which may be present are given in Tables 2 and 4. They include the Dispatch Sequence Number and the Submission Time.

An Outlog Entry is deleted from the Outlog as a whole; this deletion is only possible if all older Outlog Entries have been deleted or are deleted.

7.2.6 Auto-correlation Log

When a message or probe is dispatched, the MBC may request the MBS to automatically correlate associated "returns" delivered subsequently to the MS. The Auto-correlation Log maintains a record of these requests and the results of any auto-correlation with incoming messages that has been performed, including the MS Sequence Number of such messages. Each Auto-correlation Log entry records the requests and returns for a particular dispatch, identified by both its Dispatch Sequence Number and the Submission Time.

Each possible type of return is defined by its Return Class. An auto-correlation request is expressed in terms of the Return Classes for which auto-correlation is required. The Return Classes which are defined in this Standard are:

- for the General Mailbox Service (clause 9.2):
  - Delivery/Non-delivery Notifications

- for the IPM-specific Mailbox Service (clause 9.3):
  - Receipt/Non-receipt Notifications
  - Replies

For each Return Class, the correlation is performed with respect to one data element, known as the Auto-correlation element. For the above Return Classes:

- Delivery/Non-delivery Notifications are correlated with respect to those recipients in the envelope for which the MBC has requested such a notification.
- Receipt/Non-receipt Notifications are correlated with respect to those recipients (of whichever type) in the heading of the IP-message for which the MBC has requested such a notification. In addition for an IP-message for which auto-correlation for this class is requested, a record is maintained of all incoming Receipt/Non-Receipt Notifications which have not been requested.
- Replies are correlated with respect to those recipients (of whichever type) in the heading of the IP-message for which the MBC has requested a reply. In addition an IP-Message for which auto-correlation for this class is requested, a record is maintained for all incoming replies which have not been requested.

For each expected Notification or Reply for which auto-correlation has been requested, a Return Status is maintained in the Auto-correlation Log.

Expected Notifications can either be positive or negative; therefore each of the corresponding Return Status values is, at any moment in time, one of the following:

- negative return not received: only a negative return is expected and it has not yet been received;

- return not received: a return is expected, be it positive or negative, and it has not yet been received;

- negative return received

- positive return inferred: only a negative return is expected, but a positive return is inferred as no negative return has been received within a certain time-out limit. These limits are an implementation matter (and could differ for different message priorities, for example) but are typically expressed in hours or days rather than in seconds;

- positive return received.

For expected Replies each Return Status has, at any moment in time, one of the following values:

- reply received;

- reply not received.

For unexpected Notifications and Replies the Return Status has the value:

- unexpected return received.

A complete Auto-correlation Log Entry therefore consists of the following:

- the Dispatch Sequence Number;

- the Submission Time;

- information for each Return Class requested:

  - the identifier of the Return Class;

  - and for each instance of the auto-correlation element (both for expected and unexpected instances)

    - the elements' value;

    - the Return Status;

    - the MS Sequence Number of the associated return, where relevant.

An Auto-correlation Log Entry also maintains a synopsis of this information for each Return Class consisting of the number of returns (received or expected as appropriate) by Return Status.

An Auto-correlation Entry is deleted from the Auto-correlation Log as a whole, independently of the deletion of any other Auto-correlation Log Entry.

7.2.7 Actions

During its existence, an MS Message is the subject of Actions, either explicitly requested by the MBC, or performed automatically by the MBS on behalf of the MBC. Requests for automatic Actions are registered by the MBC in advance. These registration requests are not retrospective.

A registration parameter may be used in connection with registration requests. It contains information required to perform the automatic Action.

The Action History Attribute may be used to keep a record of the Actions that have been performed on an MS Message.

An action parameter is contained in the Action History Attribute. It contains information that resulted from the execution of the Action.

The Actions that may be performed can be classified as General Actions and IPM-specific Actions, according as they relate to General Mailbox Services (clause 9.2) and IPM-specific Mailbox Services (clause 9.3). The Actions themselves are summarised in clause 9.6.

7.2.8 Filters

Some of the functions that are made available by the MBS on Mailbox Objects allow the retrieval of a set of entries from the Object. Normally a facility is offered with these functions that allows the MBC to indicate that only those entries should be returned that meet certain criteria. This facility is known as filtering and the parameter that specifies the criteria to be met is called the filter.

A filter allows the MBC to specify a boolean expression of Attributes and Attribute values that must be met by the entries that are to be returned to the MBC.

Filters are described in more detail in clause 8.5.

8. MAILBOX ATTRIBUTES AND FILTERS

8.1 Classification of Mailbox Attributes

The Mailbox Attributes which may be present in the Mailbox Objects can be classified as General Mailbox Attributes and IPM-specific Mailbox Attributes.

The General Mailbox Attributes are those that are insensitive to the message content type, and do not convey information that is internal to the content.

The IPM-specific Mailbox Attributes are those that convey information that is specific to Interpersonal Messaging. The MBS must have knowledge of the internal structure of an IPM message (P2) in order to support these Attributes. This extension is the first of a potential future family of extensions specialized for different content types. This is illustrated in Figure 4.

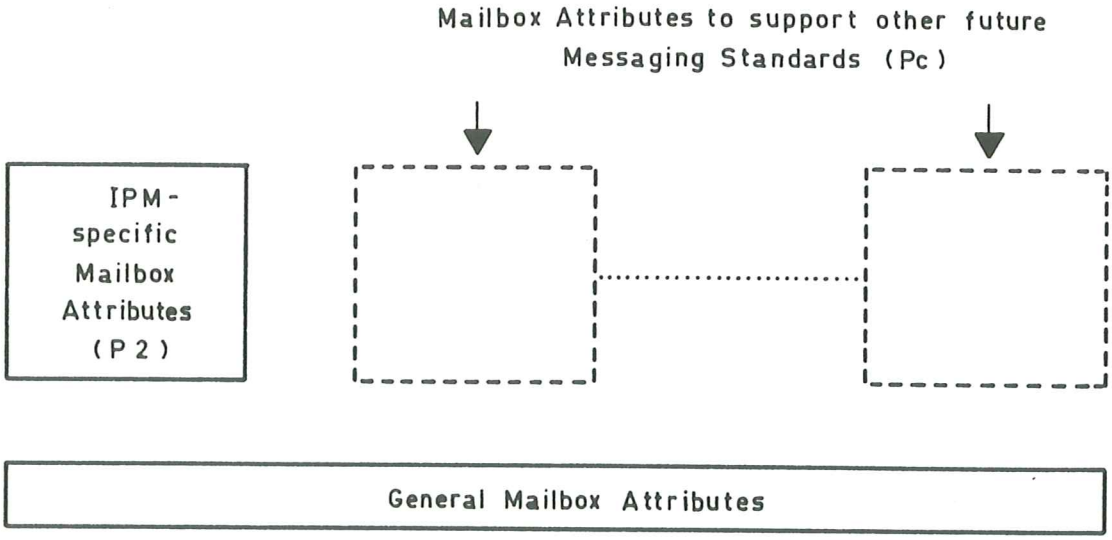

Figure 4 : Model of classification of Mailbox Attributes

8.2 General Mailbox Attributes

The General Mailbox Attributes that may occur in an MS Entry or in an Inlog Entry are listed in Table 1. They are constructed mainly from the parameter information from the DELIVER or NOTIFY primitives as defined in ECMA-93, and such Attributes are correspondingly named.

The General Mailbox Attributes that may occur in an Outlog Entry are listed in Table 2. They are constructed mainly from the parameter information from the SUBMIT or PROBE primitives as defined in ECMA-93, and such Attributes are correspondingly named.

For most Attributes, the semantics and conditionality are as already defined (in clause 7.2 or in ECMA-93). For the other Attributes, they are defined below :

Note 3

No Attributes are defined for the Submit, Probe or Deliver Event IDs as an MBC identifies events by the Dispatch or MS Sequence Numbers. No Attribute is defined for the UA Content ID as the MBS uses this parameter for its own purposes (to convey the Dispatch Sequence Number) and does not make it available to the MBC.

Action History

This Attribute describes the Actions (clause 7.2.7) that have been successfully performed on the MS Message to which this Attribute relates. It consists of the following set of information for each Action:

- date and time of the Action;
- type of the Action (clause 9.2 for General Actions and clause 9.3 for IPM-specific Actions);
- user who initiated the Action (only for a non-automatic Action);
- parameter associated with the Action (see Tables 7 and 8);
- parameter associated with the registration of the request for the Action (only for an automatic Action) (see Tables 7 and 8).

Auto-correlated

This Attribute indicates whether the message to which this Attribute relates has been used by the MBS for auto-correlation. This Attribute is not available if Auto-correlation services are not supported by the MBS.

Auto-correlation Request

This Attribute records the Return Classes for which auto-correlation has been requested. It is not available if Auto-correlation services are not supported by the MBS.

Cancelled

This Attribute indicates whether the Dispatch Deferred Delivery Cancellation service has been invoked.

Content Length

This Attribute gives the estimated length in octets of the content of a message being probed, or the length of the content of a submitted or delivered message, or the length of the content returned with a non-delivery notification. When the length of the content cannot be precisely determined, an estimated value is given. Where none of these apply, this Attribute is absent.

Content Returned

This Attribute indicates whether a content has been returned with a Non-delivery Notification.

Converted

This Attribute indicates whether the content of the MS Message has been converted.

Delivered EITs

This Attribute identifies the encoded information types in the content of the MS Message as delivered. It is the original encoded information types if the content was not converted and the converted encoded information types if it was.

Delivery Type

This Attribute indicates whether the MS Message is a delivered message or a notification.

Dispatch Type

This Attribute indicates whether a probe or a message was dispatched.

Per Message Flag

This Attribute contains the following parameters from the SUBMIT or PROBE primitives:

- Disclose-recipients
- Conversion-prohibited
- Alternate-recipient-allowed
- Content-return
- Redirection-prohibited

Recipient Info

This Attribute contains the following parameters from the SUBMIT or PROBE primitives:

- Recipient-O/R-name
- NDN-suppress
- Delivery-notice
- Explicit-conversion
- Alternative-recipient

User

This Attribute identifies the Mailbox User who invoked the dispatch.

The key to Tables 1-4

Support level by the MBS:

M = Mandatory

O = Optional

Presence of the Attribute:

P = always present in the entry, because:

- it is mandatory generated by the MBJ, or
- it is mandatory parameter in the relevant service primitive, or
- it is included as a mandatory parameter in the protocol (e.g. in the IPM Header)

C = conditionally present in the entry. It would be present because:

- it is supported by the Mailbox Server and subscribed to by the Mailbox Owner and :
- it was present in an optional parameter in the relevant service primitive, or
- it was present in an optional parameter in the protocol.

\- = always absent, otherwise.

Attribute	Multi-valued	Support level by MBS	Presence of the Attribute		Used for filter-ing	Used for Synop-sis
			Delivered Message	Notifi-cations		
Action History	Y	O	P	P	N	N
Auto-correlated	N	O	P	P	Y	Y
Content Length	N	O	P	C	Y	Y
Content Returned	N	O	-	P	Y	Y
Content Type	N	M	P	-	Y	Y
Converted	N	O	P	-	Y	Y
Delivered EITs	N	O	P	-	N	N
Delivery Time	N	M	P	P	Y	Y
Delivery Type	N	M	P	P	Y	Y
Dispatch Seq. Number	N	O	-	P	Y	Y
Intended Rec. O/R Name	N	O	C	-	Y	N
MS Sequence Numb.	N	M	P	P	Y	Y
MS Status	N	M	P	P	Y	Y
Originator O/R Name	N	O	P	-	Y	N
Original EITs	N	O	P	-	N	N
Other Recipient O/R Name	Y	O	C	-	Y	N
Priority	N	O	P	-	Y	Y
Reassignment Info	Y	O	C	-	N	N
Submission Time	N	O	P	-	Y	Y
This Recipient O/R Name	N	O	P	-	Y	N

TABLE 1 : The Message Store and Inlog General Mailbox Attributes.

Attribute	Multi-valued	Support level by MBS	Presence of the Attribute		Used for filter-ing	Used for Synop-sis
			Delivered Message	Notifi-cations		
Auto-correlation Request	Y	O	C	C	Y	N
Cancelled	N	O	P	-	Y	Y
Content Length	N	O	P	C	Y	Y
Content Type	N	M	P	P	Y	Y
Deferred Delivery Time	N	O	C	-	Y	Y
Dispatch Sequence Number	N	M	P	P	Y	Y
Dispatch Type	N	M	P	P	Y	Y
Latest Delivery Time	N	O	C	-	Y	Y
Original EITs	N	O	C	C	N	N
Originator O/R Name	N	O	P	P	Y	N
Per Message Flag	N	O	P	-	N	N
Priority	N	O	P	-	Y	Y
Recipient Info	Y	O	P	P	N	N
Recipient O/R Name	Y	O	P	P	Y	N
Submission Time	N	M	P	P	Y	Y
User	N	O	P	P	Y	Y

TABLE 2 : The Outlog General Mailbox Attributes.

8.3 IPM-specific Mailbox Attributes

The IPM-specific Attributes that may occur in an MS Entry or in an Inlog Entry are listed in Table 3. They are constructed mainly from the fields of the IP-message Heading or of the IPM-status-report (the IP-message or the IPM-status-report represents the MS Content of a delivered message), and such Attributes are correspondingly named.

The IPM-specific Attributes that may occur in an Outlog Entry are listed in Table 4. They are constructed mainly from the fields of the IP-message Heading or of the IPM-status-report which was dispatched and submitted.

For most Attributes, the semantics and conditionality are as already defined (in clauses 7.2 or in ECMA-93). For the other Attributes, they are defined below.

IPM Content Returned

This Attribute indicates whether a content has been returned with a Non-Receipt Notification.

IPM Message Type

This Attribute indicates whether an IPM message in an IP-message, a Receipt Notification or a Non-receipt Notification.

IPM Synopsis

This Attribute shows the internal structure of an IP-message and the access that has been made to the individual Body Parts by Users. It is created and later maintained by the MBS.

Attribute	Multi-valued	Support level by MBS	Presence of the Attribute		Used for filter-ing	Used for Synop-sis
			Delivered Message	Notifi-cations		
Authorizing User	Y	O	C	-	Y	N
Auto-forwarded	N	O	P	-	Y	Y
Blind Copy Recipient	Y	O	C	-	Y	N
Copy Recipient	Y	O	C	-	Y	N
Cross Reference	Y	O	C	-	Y	N
Expiry Date	N	O	C	-	Y	Y
Importance	N	O	P	-	Y	Y
In Reply To	N	O	C	-	Y	N
IPM Content Returned	N	O	-	P	Y	Y
IP-message Id	N	M	P	P	Y	N
IPM Message Type	N	M	P	P	Y	Y
IPM Synopsis	N	M	P	-	N	N
Non-receipt Information	N	O	-	C	N	N
Obsoletes	Y	O	C	-	Y	N
Obsoleting Time	Y	O	C	-	Y	N
Originator (from Heading)	N	O	C	-	Y	N
Primary Recipient	Y	O	C	-	Y	N
Receipt/Non-rec. Notification	N	M	-	P	Y	N
Receipt Info.	N	O	-	C	N	N
Reply By	N	O	C	-	Y	Y
Reply To User	Y	O	C	-	Y	N
Sensitivity	N	O	C	-	Y	Y
Subject	N	O	C	-	Y	Y

TABLE 3 : The Message Store and Inlog IPM-specific Mailbox Attributes

Attribute	Multi-valued	Support level by MBS	Presence of the Attribute		Used for filter-ing	Used for Synop-sis
			Delivered Message	Notifi-cations		
Authorizing User	Y	O	C	-	Y	N
Auto-forwarded	N	O	P	-	Y	Y
Blind Copy Recipient	Y	O	C	-	Y	N
Copy Recipient	Y	O	C	-	Y	N
Cross Reference	Y	O	C	-	Y	N
Expiry Date	N	O	C	-	Y	Y
Importance	N	O	P	-	Y	Y
In Reply To	N	O	C	-	Y	N
IP-message Id	N	M	P	P	Y	N
IPM Message Type	N	M	P	P	Y	Y
Non-receipt Information	N	O	-	C	N	N
Obsoletes	Y	O	C	-	Y	N
Obsoleting Time	Y	O	C	-	Y	N
Originator (from Heading	N	O	C	-	Y	N
Primary Recip.	Y	O	C	-	Y	N
Receipt/Non-rec. Notification	N	M	-	P	N	N
Receipt Info.	N	O	-	C	N	N
Reply By	N	O	C	-	Y	Y
Reply To User	Y	O	C	-	Y	N
Sensitivity	N	O	C	-	Y	Y
Subject	N	O	C	-	Y	Y

TABLE 4 : The Outlog IPM-specific Mailbox Attributes

8.4 The Handling of unrecognized Attributes

When an MBC retrieves information from the MBS, the MBS may send some (optional) Attributes to the MBC, which have been subscribed to but the MBC does not recognize. In such cases the MBC ignores the unrecognizable Attributes and only uses the ones it has subscribed to. The decision whether the existence of such Attributes is conveyed to the User is left to the implementor.

8.5 Filters

A filter consists of a selection criterion or a logical combination of them.

The selection criteria are:

- equality of the Attribute value;
- substrings, for further study;
- greater or equal, applicable to Attributes which have values that have natural orders (integers and times);
- less or equal, applicable to Attributes which have values that have natural orders;
- the presence of the specified Attribute type.

When a selection criterion is applied to a multi-valued Attribute, the selection criterion yields TRUE if the condition is TRUE for at least one of the Attribute values.

When combining selection criteria, logical "anding" and "oring" of individual criteria or their logical negations, is allowed. However, combining together "and's" and "or's" into a filter is not allowed.

9. MAILBOX SERVICE DESCRIPTION

This clause describes the services provided to a Mailbox Client (MBC) by a Mailbox Server (MBS). Some of these services are described in ECMA-93 MIDA and are referenced from this clause; the services additional to those described in that Standard are described here.

9.1 Overview

The services provided by an MBS to an MBC can be classified as General Mailbox Services and Mailbox Services specific to Interpersonal Messaging.

The General Mailbox Services are insensitive to the message content type, and treat the content as an indivisible unit. This allows the MBC, for example, to fetch any content type, including undefined or unknown ones.

The services which are specific to Interpersonal Messaging have knowledge of the internal structure of an IP-message content (P2) e.g. heading, nested body parts etc, and the IPM services specific to P2 e.g. forwarding and receipts. This extension is the first of a potential future family of extensions specialized for different content types. This is illustrated in Figure 5.

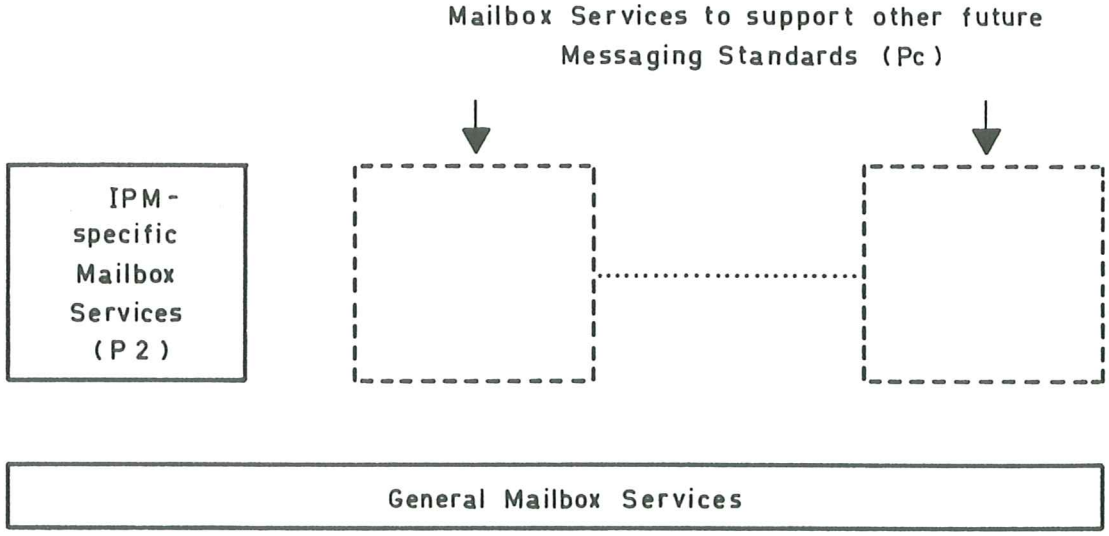

Figure 5 : Model of classification of Mailbox Services

9.1.1 Classification of Services

The following General Mailbox Service Groups from the MIDA Message Transfer Service are made available to the MBC by the MBS (see Table 5):

- Basic Message Transfer (except for Access Management);
- Submission and Delivery;
- Conversion;
- Query.

The details of these services are described in Standard ECMA-93 MIDA.

The "Probe" Service of ECMA-93 is not available as an equivalent service is offered by the "Dispatch Probe" Service.

Note that the classification of Services into Service Groups is not fully equivalent to the classification in ECMA-93.

Additional General Mailbox Service Groups defined in this Standard are:

- Basic Mailbox;
- Dispatch;
- Message Store;
- Inlog;
- Outlog;
- Auto-correlation Request;
- Auto-correlation Information;
- Registration (of Automatic Actions);
- Third Party Transfer;
- Status and Inform.

The following IPM-specific Mailbox Service Groups from the MIDA Interpersonal Messaging are made available to the MBC by the MBS (see Table 6):

- IPM Basic;
- Cooperating IPM UA - Action;
- Cooperating IPM UA - Information Conveying.

The details of these services are described in Standard ECMA-93 MIDA.

Additional IPM-specific Mailbox Service Groups defined in this Standard are:

- IPM Dispatch;
- IPM Message Storage;
- IPM Auto-correlation Request;
- IPM Service Activation/Deactivation.

9.2 General Mailbox Services

This clause contains a detailed description of those General Mailbox services and service groups that are not described in ECMA-93 MIDA. Each service group contains a set of services which make particular mailbox services available to the mailbox client (MBC).

9.2.1 Basic Mailbox Services (Mandatory)

9.2.1.1 Bind

This service allows a binding to be established between a Mailbox Client (MBC) and the User and, on the other hand, a Mailbox Server.

In the binding process the Mailbox to be accessed is specified and security parameters may be exchanged between the MBC (and the User) and the Mailbox Server. These security parameters may serve a number of purposes including authentication, access control, etc.

The scope and semantics of these parameters will be defined by an actual security policy, formulated and implemented by the organization operating the Mailbox application.

In the simplest case there is no specific security policy (all are trusted parties using trusted facilities) and the security parameter set in the Bind Specification will be empty.

Another extreme example allows a User to present a (security) capability and a challenge. Both are passed to the Mailbox Server in the Bind processed and results are returned to the User allowing him to authenticate the Mailbox Server on the basis of the response to the challenge.

It should be noted that the authentication of the User (including the human user) may take place external to the Mailbox application; resulting access control attributes may be used in the Bind process to allow the Mailbox Server to make an access control decision.

Either the MBC (for the User) or the Mailbox Server may abort the Bind process if the security parameters do not justify successful completion of the Bind.

The MBC can also enter fetch restrictions that are imposed for the duration of the binding, unless specifically overridden. These would be used if, for this binding, the capabilities of the Mailbox Client are less than the registered Mailbox restrictions. The restrictions concern:

- Content Types;
- Encoded Information Types;
- Maximum Content Length.

The MBC proposes one of the conformance subsets to apply to the current binding. If the MBS cannot support the proposed subset the binding will not be established.

9.2.1.2 Unbind

This service allows an MBC to terminate the current binding.

9.2.2 Dispatch Services

These services enable an MBC to dispatch a message to the MBS for submission to the MTSL. If the message is dispatched for deferred delivery, the message can be cancelled before the MTA relays the message.

9.2.2.1 Dispatch Message (Mandatory)

This service enables the MBC to dispatch a message of any content type to the MBS. The dispatched message is directly passed on for submission to the MTSL, even if Deferred Delivery is requested. The content may be supplied explicitly or, optionally, by a reference for the Third Party Transfer service (see clause 9.2.9). Auto-correlation may be requested (see clause 9.2.6). In turn, the MBS provides the MBC with the Dispatch Sequence Number of the dispatched message and the Submission Time.

9.2.2.2 Dispatch Returned Message (Optional)

This service enables an MBC to dispatch a message with a content which was returned in a Non-Delivery Notification. The MBC shall specify new values for the submission parameters. The message is directly passed for submission to the MTSL, even if Deferred Delivery is requested. Auto-correlation may be requested. In turn, the MBS provides the MBC with the Dispatch Sequence Number of the dispatched message and the Submission Time.

9.2.2.3 Dispatch Probe (Optional)

This service enables the MBC to dispatch a probe to the MBS. The probe is directly passed to the MTSL. Auto-correlation may be requested (see clause 9.2.6). In turn, the MBS provides the MBC with the Dispatch Sequence Number of the dispatched message and the Submission Time.

9.2.2.4 Dispatch Deferred Delivery Cancellation (Optional)

This service enables an originating MBC to instruct its associated MTA, via the MBS, to cancel a previously successfully submitted message for deferred delivery. The cancellation attempt may not always succeed. Possible reasons for failure are: deferred delivery time has passed, or the message has already been relayed within the MTS.

The message to be cancelled is identified by its Dispatch Sequence Number.

9.2.3 Message Store Services

These services provide access to the Message Store.

9.2.3.1 Request Message Store Synopsis (Mandatory)

This service enables the MBC to obtain counts of selected messages in the Message Store analyzed by the values taken by a chosen Attribute or pair of Attributes present in the MS Entries of the messages. For this purpose, the absence of an available Attribute in an MS Entry is considered to be a special value. Table 1 indicates which Attributes may be chosen for analysis.

The primary criterion for selecting the MS Messages to be analysed is by specifying a range (possibly open-ended) delimited by MS Sequence Number and/or Delivery Time. Within this range, a further selection may be made by filtering based upon the values of Attributes in the messages' MS Entries.

The service also enables the MBC to obtain the lowest and highest MS Sequence Numbers of the MS Messages so selected and also their total count.

9.2.3.2 List Message Store Messages (Mandatory)

This service enables the MBC to obtain an extract from the MS Entries of selected MS Messages in the Message Store. The primary selection is by specifying a range of messages (possibly open-ended) delimited by MS Sequence Number and/or Delivery Time. Within this range, a further selection may be made based upon the values of Attributes in the messages' MS Entries.

For each selected message, all available Attributes in the MS Entry are returned unless the MBC specifies a list of the Attribute Types that should be returned.

If the MS Status of an MS Message starts as "New" or "Listed", it shall end as "Listed". If the MS Status is "Processed", it remains as "Processed".

The value of the MS Status Attribute, if returned, is that prior to invoking this service.

9.2.3.3 Fetch Message Store Message (Mandatory)

This service enables the MBC to fetch an MS Message from the Message Store. The MS Message to be fetched may be specified by its MS Sequence Number. Alternatively, the MS Message fetched is the oldest one satisfying specified selection criteria, in which case a list of the MS Sequence Numbers of further messages satisfying the selection criteria is also returned.

The primary criterion for selecting the MS Message is by specifying a range of messages (possibly open-ended) delimited by MS Sequence Number and/or Delivery Time. Within this range, a further selection may be made based upon the values of Attributes stored in the messages' MS Entries.

The MBC may specify which parts of the MS Message are to be fetched, the parts being the MS Entry, the MS Envelope and the MS Content (see clause 7.2).

The current fetch restrictions are applied when fetching the content (see clause 9.2.1.1). If the message content does not fall within the limits of the fetch restrictions, an error is returned unless an explicit override has been specified in the fetch request (in which case the complete content is returned).

The MS Status of the fetched MS Message may start as "New", "Listed" or "Processed". For a Delivery Report, it is changed to "Processed" if the MS Envelope is fetched; for a User Message, it is changed to "Processed" if the MS Content is fetched. If the MS Status starts as "New", it is at least changed to "Listed".

The value of the MS Status Attribute, if returned, is that prior to invoking this service.

9.2.3.4 Delete Message Store Messages (Mandatory)

This service enables the MBC to delete one or more MS Messages from the Message Store by specifying one or more MS Sequence Numbers or a range (possibly open-ended)

delimited by MS Sequence Number and/or Delivery Time. An MS Message may not be deleted if its MS Status is "New".

9.2.4 Inlog Services (Optional)

These services provide access to the Inlog.

Although the Inlog Services are optional, either the whole group or none of the services must be implemented.

9.2.4.1 Request Inlog Synopsis

This service enables the MBC to obtain counts of selected Inlog Entries analysed by the values taken by a chosen Attribute or pair of Attributes present in the Inlog Entries. For this purpose the absence of an available Attribute in an Inlog Entry is considered to be a special value. Table 1 indicates which Attributes may be chosen for the analysis.

The primary criterion for selecting the Inlog Entries to be analysed is by specifying a range (possibly open-ended) delimited by MS Sequence Number and/or Delivery Time. Within this range, a further selection may be made by filtering based upon the values of Attributes in the Inlog Entries.

The service also enables the MBC to obtain the lowest or highest MS Sequence Numbers of the Inlog Entries so selected and also their total count.

9.2.4.2 List Inlog Entries

This service enables the MBC to obtain selected entries from the Inlog. The primary selection is by specifying a range of entries (possibly open-ended) delimited by MS Sequence Number and/or Delivery Time. Within this range of entries, a further selection may be made based upon the values of Attributes stored there.

For each selected entry, all the available Attributes in the entry are returned unless the MBC specifies a list of the Attribute Types that should be returned.

If the MS Status of an MS Message whose Inlog Entry is listed starts as "New" or "Listed", it shall end an "Listed"; otherwise it remains unchanged.

The value of the MS Status Attribute, if returned, is that prior to invoking this service.

9.2.4.3 Delete Inlog Entries

This service enables the User to delete a sequence of Inlog Entries by specifying the upper limit MS Sequence Number or Delivery Time. Entries are deleted starting from the entry with the lowest MS Sequence Number. A Mail Inlog entry may only be deleted if the corresponding MS Message has been deleted. The delete is only performed if all the Inlog entries in the sequence can be deleted.

9.2.5 Outlog Services (Optional)

These services provide access to the Outlog.

Although the Mail Outlog Services are optional, either the whole set or none of the services must be implemented.

9.2.5.1 Request Outlog Synopsis

This service enables the MBC to obtain counts of selected Outlog Entries analysed by the values taken by a chosen Attribute or pair of Attributes present in the Outlog Entries. For this purpose the absence of an available Attribute in an Outlog Entry is considered to be a special value. Table 2 indicates which Attributes may be chosen for analysis.

The primary criterion for selecting the Outlog Entries to be analysed is by specifying a range (possibly open-ended) delimited by Dispatch Sequence Number and/or Submission Time. Within this range, a further selection may be made by filtering based upon the values of the Attributes in the Outlog Entries.

The service also enables the MBC to obtain the lowest and highest Dispatch Sequence Numbers of the Outlog Entries so selected and also their total count.

9.2.5.2 List Outlog Entries

This service enables the MBC to obtain selected entries from the Outlog. The primary selection is by specifying a range of entries (possibly open-ended) delimited by Dispatch Sequence Number and/or Submission Time. Within this range of entries, a further selection may be made based upon the values of Attributes stored there.

For each selected entry, all available Attributes in the entry are returned unless the MBC specifies a list of the Attribute Types that should be returned.

9.2.5.3 Delete Outlog Entries

This service enables the User to delete a sequence of Outlog Entries by specifying the upper limit Dispatch Sequence Number or Submission Time. Entries are deleted starting from the entry with the lowest Dispatch Sequence Number.

9.2.6 Auto-correlation Request Service (Optional)

This service provides the possibility to request automatic correlation of returns for dispatched messages or probes and the maintenance of an Auto-correlation Log in which the correlation information is held.

The Auto-correlation Request service is only meaningful in combination with the Auto-correlation Information Services.

9.2.6.1 Request Auto-correlation of Delivery and Non-delivery Notifications (Optional)

This service enables the MBC at the time of dispatch to request the MBS to automatically correlate any Delivery and Non-delivery Notifications for the message or probe being submitted to the MTSL.

Any MS Message delivered to the MS and containing Delivery or Non-delivery Notifications will be given an MS Status of "processed" when these notifications are correlated by the MBS. The results of such auto-correlations are placed in the Auto-correlation Log Entry for the dispatched message. Deletion of the log entry implies the deletion of all the MS Messages containing the associated Delivery and Non-delivery Notifications.

9.2.7 Auto-correlation Information Services (Optional)

These services perform a requested automatic correlation of returns for previously dispatched messages or probes and the maintenance of an Auto-correlation Log in which the correlation information is held. The Return Classes for which these services may be provided and the Auto-correlation Log are both defined in clause 7.2.6.

Although the Auto-correlation Information Services are optional, either the whole set or none of the services must be implemented. Either the Auto-correlation Request service or the IPM-specific Auto-correlation Request services must be used in combination with the Auto-correlation Information services, in order to make the latter meaningful.

9.2.7.1 List Auto-correlation Entries

This service enables the MBC to obtain a list of stored Auto-correlation Log Entries, specified by a range delimited by Dispatch Sequence Number and/or Submission Time.

9.2.7.2 Read Auto-correlation Entry

This service enables the MBC to retrieve the Auto-correlation Log Entry for a given message by specifying its Dispatch Sequence Number.

9.2.7.3 Delete Auto-correlation Entry

This service enables the MBC to delete the Auto-correlation Log Entry for a given message by specifying its Dispatch Sequence Number. Activation of this service implies deletion of the corresponding Delivery/Non-delivery Notifications, and Receipt/Non-receipt Notifications, but not Replies, from the Message Store. This function is executed even if not all expected returns have arrived; any returns which are subsequently received are processed as if auto-correlation had not been requested.

9.2.8 Registration Service (Optional)

9.2.8.1 Registration of Automatic Actions

This service provides the MBC with the general ability to instruct the MBS to automatically perform certain Actions on its behalf. This service also enables the MBC to terminate such an instruction and to deactivate the automatic Action.

9.2.9 Third Party Transfer Service (Optional)

The Third Party Transfer Application is not yet defined in detail. However, the Mailbox service definition has made provisions so as to be able to use the Third Party Transfer option as soon as the corresponding Application is fully specified. The Third Party Transfer option allows references to objects such as messages or lists to be returned as an alternative to returning the complete object.

9.2.10 Basic Message Transfer Services

The following Basic Message Services are defined in addition to ECMA-93.

9.2.10.1 Registered Encoded Information Type Capability (Mandatory)

This service enables the MBC to inform the MTS via the MBS of the encoded information type(s) that can be delivered to the MBS.

9.2.10.2 Registered Content Type Capability (Mandatory)

This service allows the MBC to indicate to the MTS via the MBS what message content type may be delivered to the MBS. The content types to be supported by UAs can be ISO-defined, CCITT-defined, nationally defined, or privately defined by particular organizations. The MTS will only deliver a message to the MBS if the message uses one of the content types registered for that MBS.

9.2.10.3 Registered Content Length Capability (Mandatory)

This service allows the MBC to inform the MTS via the MBS of the maximum length of the messages that can be delivered to the MBS.

9.2.11 Status and Inform Services

The following Status and Inform Services are defined in addition to ECMA-93.

9.2.11.1 Alternate Recipient Assignment (Optional)

This service enables an MBC to be given the capability of having certain messages delivered to its MBS for which there is not an exact match between the recipient attributes specified and the descriptive name(s) of the Mailbox. Such a Mailbox 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 organization may establish a Mailbox 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 the MIDA in that organization. This permits the organization to manually handle the messages to these individuals.

In order for a message to be assigned to an alternative recipient, the originator must have requested the Alternat Recipient Allowed service element.

Messages will be delivered to the Mailbox if all of the following conditions hold :

- The originator requested the Alternate Recipient Allowed service element;
- The recipient O/R name contains a full set of attributes;
- Those attributes that are required to match do;
- Those attributes that are not required to match do not match those of any UA or Mailbox in that MD.

9.2.11.2 Hold for Delivery (Optional)

This service enables a recipient MBC to request that the MTS holds messages for delivery and returning notifications to its MBS until a later time. The MBC can indicate to the MTS when its MBS 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.

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

Note 4

This service element may have an impact upon the expected time period for delivery of a message.

The maximum time that a returning notification will be held is not specified.

The Hold for Delivery service element provides temporary storage to facilitate delivery and only after a message has been transferred to the recipient's Mailbox is delivery notification returned. This should be distinguished from Mailbox Message Store facility.

9.3 IPM-specific Mailbox Services

This clause contains a detailed description of those IPM-specific Mailbox service groups that are not described in ECMA-93 MIDA. Each service group contains a set of services which make particular Mailbox services available to the MBC.

When the IPM-Subset is used during an instance of interworking between an MBC and an MBS both General Mailbox Services and IPM-specific Mailbox services are available to the MBC.

In such a case the functionality of the General Mailbox services may exceed the functionality as defined in clause 9.2 in the following areas :

- Attributes manipulated by General Mailbox services include the IPM-specific Attributes.
- Actions manipulated by the General Mailbox services include the IPM-specific Actions.
- Return Classes manipulated by the General Mailbox services include the IPM-specific Return Classes.

The classification of IPM-specific services as Mandatory and Optional is only relevant when conformance to the IPM Subset is claimed (see clause 3.6).

9.3.1 IPM Dispatch Services

9.3.1.1 Dispatch IP-message (Mandatory)

This service enables the MBC to dispatch an IP-message to the MBS. The IP-message is directly passed on for submission to the MTSL, even if Deferred Delivery is requested. In response to the IP-message Dispatch, the MBS returns a Submission Time and a Dispatch Sequence Number.

The message dispatched to the MBS consists of an envelope, a heading, and a body which itself may consist of several body parts.

The IP-message dispatched by the MBC may be incomplete, as follows:

- If the IPMessageId or Originator components are omitted from the heading, then these will be generated by the MBS.
- If the EncodedInformationTypes component is omitted from the envelope, then this will be generated by the MBS. (Note that the MBS is unable to do this if the body consists of a single TTX or G3Fax body part and the associated non-basic parameters have been omitted from that body part).
- If the list of Recipients component is omitted from the envelope, then this may be extracted from the heading by the MBS, in which case the relevant O/R Descriptors must contain an O/R Name.

9.3.1.2 IP-message Reply Instruction (Optional)

This service enables the MBC to provide a message to the MBS as a response to a IP-message still stored in the Mailbox. In turn, the MBS provides the MBC with the Submission Time, and the Dispatch Sequence Number of the dispatched message.

The referenced IP-message is used by the MBS to construct parts of both the P2 Heading and the P1 Envelope in the Reply IP-message.

The message provided to the MBS consists of an envelope, a heading, and a body which itself may consist of several body parts.

The IP-message provided by the MBC may be incomplete, as follows:

- If the IPMessageId or Originator components are omitted from the heading, then these will be generated by the MBS.
- If the EncodedInformationTypes component is omitted from the envelope, then this will be generated by the MBS. (Note that the MBS is unable to do this if the body does not consist of a single TTX or G3Fax body part and the associated non-basic parameters have been omitted from that body part).
- If the list of Recipients component is omitted from the envelope, then this may be extracted from the heading by the MBS, in which case the relevant O/R Descriptors must contain an O/R Name.

Several elements of the heading of the submitted IP-message will be derived by the MBSE from the heading of the IP-message being replied to if they are not present in the heading presented in the argument. This applies to:

- inReplyTo
- subject
- importance
- sensitivity

The reply is to be sent to one or more of the following groups of potential recipients:

- the originator of the original message;
- the user(s) which the originator designated that the reply should be sent to; and
- all the recipients of the original message.

Duplicates between the groups are eliminated, as is the current originator.

In addition, the User may specify O/R Names of additional Recipients of the Reply.

More than one Reply may be sent for the same message. The User that performs the Reply may, but need not, be one of the Recipients with the Reply Request Indication.

The MS Status must start as "Listed" or "Processed", and will end as "Processed".

9.3.1.3 IP-message Forwarding Instruction (Optional)

This service enables the MBC to dispatch an IP-message to the MBS, for onward submission to the MTSL; where some or all of the Body Parts of this message are constructed by the MBS from complete IP-messages, or returned IP-messages stored in the Mailbox. These are referenced by their MS Sequence Numbers. In response to the Forwarding instruction, the MBS returns a Submission Time and a Dispatch Sequence Number.

The forwarded IP-Message(s) form new body parts, with the body part type "Forwarded IP-Message", in the new IP-Message.

In addition to the forwarded body parts, new body parts can be included in the new IP-Message.

For each of the IP-messages being forwarded, the MS State must start as "Listed" or "Processed" and will remain unchanged.

9.3.1.4 Dispatch Returned IP-message (Optional)

This service enables a MBC to re-dispatch an IP-message which was returned in a Non-Receipt Notification and is still in the Message Store. The Non-Receipt Notification is referenced by its MS Sequence Number. In response to this Dispatch instruction the MBS returns a Submission Time and a Dispatch Sequence Number.

The returned Body forms the Body of the new IP-message.

The IP-message dispatched by the MBC may be incomplete, as follows:

- If the IPMessageId or Originator components are omitted from the heading, then these may be generated by the MBS.
- If the EncodedInformationTypes component is omitted from the envelope, then this may be generated by the MBS. (Note that the MBS is unable to do this if the body does not consist of a single TTX or G3Fax body part and the associated non-basic parameters have been omitted from that body part).
- If the list of Recipients component is omitted from the envelope, then this may be extracted from the heading by the MBS, in which case the relevant O/R Descriptors must contain an O/R Name.

9.3.1.5 IP-message Non-receipt Instruction (Optional)

This service enables the MBC to generate a Non-receipt Notification for an IP-message by specifying the MS Sequence Number of the IP-message and the reason for the non-receipt.

The referenced IP-message is used by the MBS to construct the IPM-status report containing the non-receipt indication and, if requested, the content.

9.3.1.6 IP-message Receipt Instruction (Optional)

This service enables the MBC to generate a Receipt Notification for an IP-message by specifying the MS Sequence Number of the message.

The referenced IP-message is used by the MBS to construct the receipt notification IPM-status report.

9.3.2 IPM Message Store Services

9.3.2.1 IPM Fetch MS Message (Mandatory)

This service enables the MBC to retrieve complete or selected parts of an IP-message from the message store for IP-messages received as well as for IP-messages received in Non-Receipt Notifications. This is achieved by specifying the MS Sequence Number of the desired message along with the component parts of the IP-message that are required. The service returns the requested body part(s).

Restrictions on the types of body parts that may be returned are imposed during binding although these may be explicitly overridden by the Fetch IP-message service.

The MS Status of the fetched MS Message may start as "New", "Listed" or "Processed". For a Delivery Report, it is changed to "Processed" if the MS Envelope is fetched; for a User Message, it is changed to "Processed" if the IP-message Synopsis shows that all body parts have been fetched. If the MS Status starts as "New", it is at least changed to "Listed". The value of the MS Status Attribute, if returned, is that prior to invoking this service.

9.3.3 IPM Auto-correlation Request Services

9.3.3.1 Request Auto-correlation of Receipt and Non-Receipt Notifications (Optional)

This service enables the MBC at the time of dispatch to request the MBS to automatically correlate any Receipt and Non-receipt Notifications for the IP-message being submitted to the MTSL. This service is used in combination with the IPM dispatch services:

Dispatch IP-message,
IP-message Reply Instruction,
IP-message Forwarding Instruction, and
Dispatch Returned IP-message.

Any message delivered to the MS and containing a Receipt or Non-receipt Notification will be given an MS Status of "processed" when this notification is correlated by the MBS. The results of such auto-correlations are placed in the Auto-correlation Log Entry for the dispatched message. Deletion of the log entry implies the deletion of all the MS Messages containing the associated Receipt or Non-receipt Notification.

9.3.3.2 Request Auto-correlation of Replies (Optional)

This service enables the MBC at the time of dispatch to request the MBS to automatically correlate any Reply for the IP-message being submitted to the MTSL. This service is used in combination with the IPM dispatch services:

Dispatch IP-message,
IP-message Reply Instruction,
IP-message Forwarding Instruction, and
Dispatch Returned IP-message.

Any message delivered to the MS and containing a Replying IP-message will not be given an MS Status of "processed" when this Reply is correlated by the MBS. The results of such auto-correlations are placed in the Auto-correlation Log Entry for the dispatched message.

9.3.4 IPM Service Activation/Deactivation

9.3.4.1 Auto-Non-receipt Notification Registration (Optional)

This service enables the MBC to instruct its associated MBS to automatically generate a Non-receipt Notification, if requested by the original originator, when an IP-message is deleted (manually or automatically) from the Message Store without being processed. This service also enables the MBC to terminate such a registration.

Non-receipt Notifications are only generated by the MBS if the Originator of the IP-message requested it and Auto-Non-receipt Notification is registered. The actions of the services:

Auto-forward Registration and
Auto-expiry Registration

therefore depend on the Auto-Non-receipt Notification Registration.

9.3.4.2 Auto-Receipt Notification Registration (Optional)

This service enables the MBC to instruct its associated MBS to automatically generate a Receipt Notification, if requested by the original originator, after the IP-message has been processed. (Note however that Auto-forwarding results in a Non-receipt Notification.) This service also enables the MBC to terminate such a registration.

9.3.4.3 Auto-forward Registration (Optional)

This service enables the MBC to instruct its MBS to automatically forward all incoming IP-messages to another UA. This service also enables the MBC to terminate such a registration.

The delivered IP-message is auto-forwarded as a body part of a new IP-message. The Heading of this IP-message will be copied from the auto-forwarded IP-message except for the following:

- IP-message Id: generated by the MBS;
- Originator: an O/R Descriptor containing the O/R Name of the Mailbox;
- Primary Recipients: the O/R Descriptor specified with the Auto-forward Registration;
- Copy Recipients: none;
- Blind Copy Recipients : none;
- Auto-forwarded Indicator : set.

When an IP-message is auto-forwarded, it is deleted from the Message Store, and a Non-receipt Notification is generated if requested by the original originator. The Auto-forward Registration service allows the Client to provide a "Comment" to be included with the Non-receipt Notification.

In order to prevent "looping", an IP-message will not be auto-forwarded if it has already been auto-forwarded.

Auto-forwarding applies to IP-messages delivered after auto-forwarding has been registered; IP-messages delivered before registration will not be auto-forwarded.

9.3.4.4 Auto-expiry Registration (Optional)

This service enables the MBC to instruct its associated MBS to automatically delete expired IP-messages, if still stored by the MBS. It is an implementation detail as to exactly when expiry is detected, but it is recommended to happen within a 24 hour interval. This service also enables the MBC to terminate such a registration.

If the old IP-message has not been processed, a Non-receipt Notification is generated if requested by the original originator.

Auto-expiry applies only to IP-messages in the Message Store that have an Expiry Date in the outermost Heading.

9.4 Message Store Status

A message can exist in the message store in one of three MS Status states, and when it is deleted from the MS, the MS Status for the message as recorded in the Inlog may take on a fourth state. This is illustrated in Figures 6 and 7 which is a State Diagram for incoming messages to the Mailbox. The diagram shows how the MS Status of a stored message can be changed as a result of operations which are performed on the message, such as Listing and Fetching.

The possible values for MS Status are defined in clause 7.2.

Figure 6 indicates the state transitions that can occur in the General Subset,

Figure 7 indicates the state transitions that can occur in the IPM Subset.

-The following legend applies to Figure 6:

- "Notification" includes Receipt/Non-Receipt and Delivery/Non-Delivery.
- "Receipt Notification" and "Non-Receipt Notification" are issued only for IP-messages and if requested by the original originator, and Receipt/Non-receipt Notification Registration is enabled.
- Remaining text refers to services described in clauses 9.2 and 9.3.

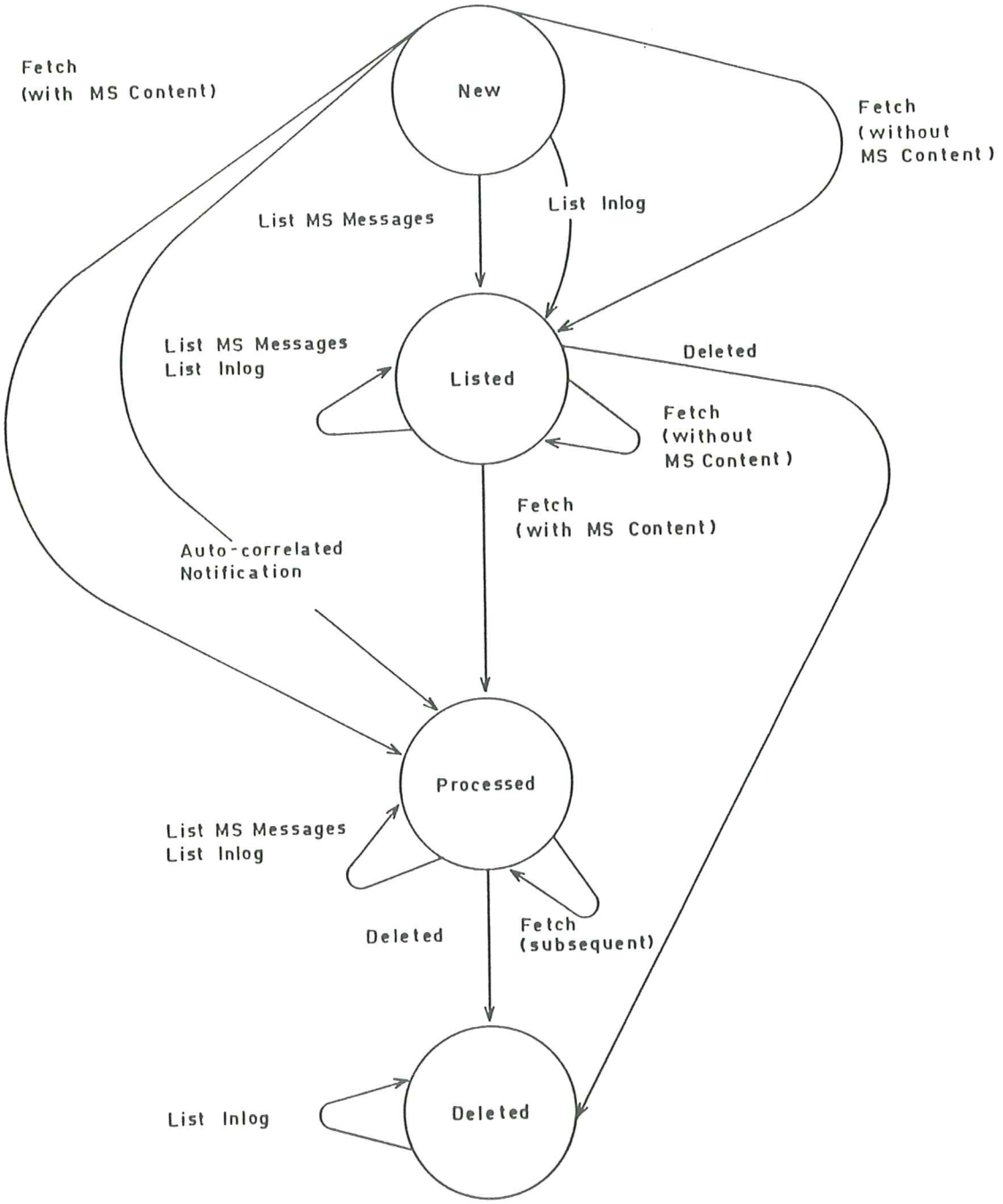

Figure 6 : State Diagram for Incoming Messages in the General Subset (MS Status)

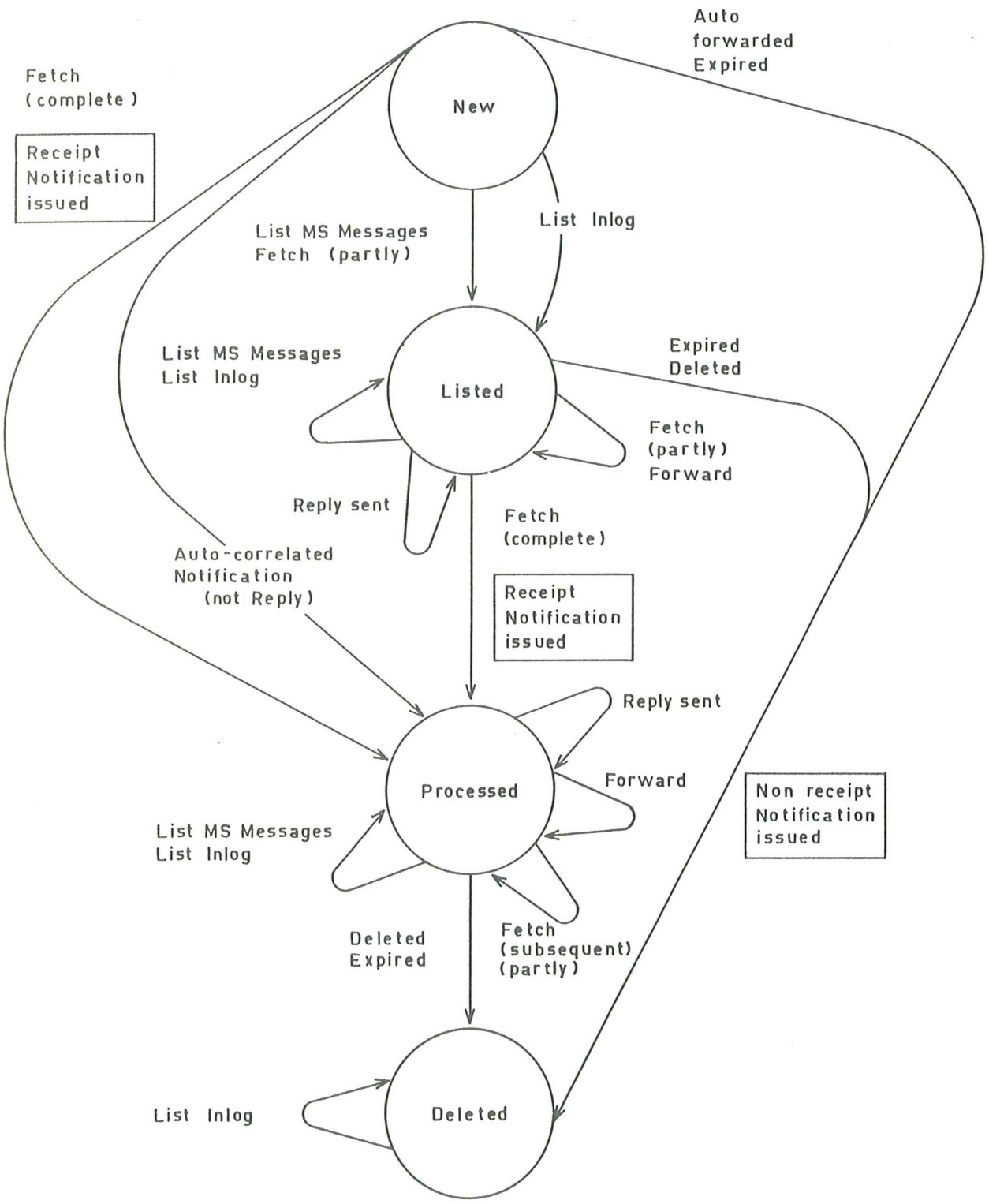

Figure 7 : State Diagram for Incoming Messages in the IPM Subset (MS Status)

9.5 Service Summary

SERVICE GROUP	SERVICES	MANDATORY	OPTIONAL
BASIC MAILBOX	Access Management (Bind/Unbind)	X	
BASIC MESSAGE TRANSFER	Content Type Indication	X	
	Converted Indication	X	
	Delivery Time Stamp Indication	X	
	Message Identification	X	
	Non-delivery Notification	X	
	Original Encoded Information Types Indication	X	
	Submission Time Stamp Indication	X	
	Registered Encoded Information Type Capability	X	
	Registered Content Type Capab.	X	
	Registered Content Length Capability	X	
SUBMISSION AND DELIVERY	Alternate Recipient Allowed	X	
	Deferred Delivery	X	
	Delivery Notification	X	
	Disclosure of Other Recipients	X	
	Grade of Delivery Selection	X	
	Multi-destination Delivery	X	
	Prevention of Non-delivery Notification		X
	Return of Contents		X
	Originator-requested Alternative Recipient		X
	Latest Delivery Designation		X
	Redirection of Incoming Messages		X
	Redirection Prohibition		X
CONVERSION	Conversion Prohibition	X	
	Explicit Conversion		X
	Implicit Conversion		X

TABLE 5 : Summary of General Mailbox Services (part 1 of 2)

SERVICE GROUP	SERVICES	MANDATORY	OPTIONAL
QUERY	Advice on the Encoded Information Type Capabilities of a Remote UA		X
	Advice on the Content Type Capabilities of a Remote UA		X
	Advice on the Content Length Capabilities of a Remote UA		X
STATUS AND INFORM	Alternate Recipient Assignment		X
	Hold for Delivery		X
DISPATCH	Dispatch Message	X	
	Dispatch Returned Message		X
	Dispatch Probe		X
	Dispatch Deferred Delivery Cancellation		X
MESSAGE STORE	Request Message Store Synopsis	X	
	List Message Store Entries	X	
	Fetch Message Store Message	X	
	Delete Message Store Messages	X	
INLOG	Request Inlog Synopsis		X
	List Inlog Entries		X
	Delete Inlog Entries		X
OUTLOG	Request Outlog Synopsis		X
	List Outlog Entries		X
	Delete Outlog Entries		X
AUTO-CORRELATION REQUEST	Request Auto-correlation of Deliv. and Non-delivery Notifications		X
AUTO-CORRELATION INFORMATION	List Auto-correlation Entries		X
	Read Auto-correlation Entry		X
	Delete Auto-correlation Entry		X
REGISTRATION	Registration of Automatic Actions		X
THIRD PARTY TRANSFER	Third Party Transfer Option		X

TABLE 5 : Summary of General Mailbox Services (part 2 of 2)

Table 6 below only makes the distinction between Mandatory and Optional services. Standard ECMA-93 gives a more detailed specification of which services are mandatory for Sending and which are mandatory for Reception.

SERVICE GROUP	SERVICES	MANDATORY	OPTIONAL
IPM BASIC	IP-message Receipt	X	
	IP-message Sending	X	
	IP-message Identification	X	
	Typed Body	X	
CO-OPERATING IPM UA ACTION	Blind Copy Recipient Indication	X	
	Non-receipt Notification		X
	Receipt Notification		X
	Auto-forwarded Indication	X	
CO-OPERATING IPM UA INFORMATION CONVEYING	Originator Indication	X	
	Authorising Users Indication	X	
	Primary and Copy Recipient Ind.	X	
	Expiry Date Indication	X	
	Cross-referencing Indication	X	
	Importance Indication	X	
	Obsoleting Indication	X	
	Sensitivity Indication	X	
	Subject Indication	X	
	Replying IP-message Indication	X	
	Reply Request Indication	X	
	Forwarded IP-message Indication	X	
	Body Part Encryption Indication	X	
	Multi-part Body	X	
	Timed Obsoleting Indication	X	

TABLE 6 : Summary of IPM-specific Mailbox Services (part 1 of 2)

SERVICE GROUP	SERVICES	MANDATORY	OPTIONAL
IPM DISPATCH	Dispatch IP-message	X	
	IP-message Reply Instruction		X
	IP-message Forwarding Instruction		X
	Dispatch Returned IP-message		X
	IP-message Non-receipt Instruction		X
	IP-message Receipt Instruction		X
IPM MESSAGE STORE	IPM Fetch MS Message	X	
IPM AUTO-CORRELATION REQUEST	Request Auto-correlation of Rec. and Non-receipt Notifications		X
	Req. Auto-correlation of Replies		X
IPM SERVICE ACTIVATION/ DEACTIVATION	Auto-Non-receipt Notification Registration		X
	Auto-Receipt Notific. Registration		X
	Auto-Forward Registration		X
	Auto-Expiry Registration		X

TABLE 6 : Summary of IPM-specific Mailbox (part 2 of 2).

9.6 Actions Summary

The General Actions are summarised in Table 7.

Type	Service	Automatic	Registration parameter (if automatic)	Action parameter
Fetch	Fetch MS Message	No	-	MS Status
Delete	Delete MS Messages	No	-	-
Redispatch	Dispatch Returned Message	No	-	Dispatch Seq. Number

TABLE 7 : Summary of General Actions.

The IPM-specific Actions are summarised in Table 8.

Type	Service	Automatic	Registration parameter (if automatic)	Action parameter
Non-Receipt	IP-message Non-receipt Instruction	No	-	Dispatch Seq. Number
Receipt	IP-message Receipt Instruction	No	-	Dispatch Seq. Number
Reply	IP-message Reply Instruction	No	-	Dispatch Seq. Number
Forward	IP-message Forward Instruction	No	-	Dispatch Seq. Number
IPM-fetch	IPM Fetch MS Message	No	-	MS Status
IPM-redispatch	Dispatch Returned IP-message	No	-	Dispatch Seq. Number
Auto-non receipt	Auto-Non-receipt Notific. Registr.	Yes	-	Dispatch Seq. Number
Auto-receipt	Auto-Receipt Notification Registration	Yes	-	Dispatch Seq. Number
Auto-forward	Auto-Forward Registr.	Yes	Note 5	Dispatch Seq. Number
Auto-expire	Auto-Expiry Registr.	Yes	-	-

TABLE 8 : Summary of IPM-specific Actions

Note 5

The Registration parameter for auto-forwarding comprises two components :

- the O/R descriptor of the new recipient,

- optional comments to be returned on any associated non-receipt notification.

Where the Action Parameter (in Tables 7 and 8) is a Dispatch Sequence Number, it refers to that of the message dispatched as a result of the Action. Where it is an MS Status, it refers to the MS Status of the MS Message after the Action was performed.

10. MAILBOX ACCESS PROTOCOL SPECIFICATION

10.1 Introduction

This clause specifies the Mailbox Access Protocol (P7), which enables an MBC that is remote from its MBS to access the MBS.

The MBC accomplishes the above and other objectives by invoking remote operations on the MBS. The mapping of the P7 protocol onto the Remote Operation Service is described in clause 11 of this Standard. Remote operations themselves are described in ECMA TR/31.

The protocol specification has been divided into its constituent parts as follows:

- the protocol supporting the general mailbox services (described in clause 10.2)
- the protocol supporting the IPM-specific mailbox extensions to the general services (described in clause 10.3)

10.2 Access Protocol for a General Mailbox Service

10.2.1 Introduction

The protocol supporting the General Mailbox Services is defined in the following ASN.1 modules:

- P7Operations: contains the definitions of the remote operations, the associated remote errors, and the supporting data types (described in clauses 10.2.2 to 10.2.4 and summarized in clause 10.2.5):
- P7Identifiers: contains the definitions of the object identifiers used in conjunction with mailbox access (described in clause 10.2.6 and summarized in clause 10.2.7):

10.2.2 Common Data Types

A number of data types figure in the calling sequences of many of the remote operations defined by the protocol. These data types have to do with the following:

MS Status
MS Sequence Numbers and Delivery Times
Dispatch Sequence Numbers and Submission Times
Object Request
Fetch Restrictions
Mailbox Attributes
Filters
Selectors
Return Classes

10.2.2.1 MS Status

Frequent use is made of the MSStatus data type, representing the MS Status of a message in the MS:

```
MSStatus::= INTEGER {
    new (0), listed (1), processed (2), deleted (3) }
```

10.2.2.2 MS Sequence Numbers and Delivery Times

The data type for an MS Sequence Number is defined as follows:

```
MSSequenceNumber ::= INTEGER
```

A "span" of MS Sequence Numbers in the MS or the Inlog is defined as follows:

```
MSSpan ::= SEQUENCE {
    lowest [0]  IMPLICIT MSSequenceNumber,
    highest [1]  IMPLICIT MSSequenceNumber}
```

The data type for a Delivery time is defined as follows:

```
DeliveryTime ::= Time
```

The message in the MS or the entries in the Inlog may be subsetted by specifying a "range" or "cutoff" defined by MS Sequence Number or Delivery Time :

```
MSRange ::=SEQUENCE {
    from [0] MSPosition OPTIONAL--omitted means no lower bound--,
    to [1] MSPosition OPTIONAL--omitted means no upper bound--}

MSCutoff ::=SEQUENCE {
    to [1] MSPosition OPTIONAL--omitted means no upper bound--}

MSPosition ::= CHOICE {
    number [0]  IMPLICIT MSSequenceNumber,
    time [1]  IMPLICIT DeliveryTime }
```

10.2.2.3 Dispatch Sequence Numbers and Submission Times

The data type for a Dispatch Sequence Number is defined as follows:

```
DispatchSequenceNumber ::= INTEGER
```

A "span" of Dispatch Sequence Numbers in the Outlog or the Auto-correlation Log is defined as follows:

```
DispatchSpan ::= SEQUENCE {
    lowest [0]  IMPLICIT DispatchSequenceNumber,
    highest [1]  IMPLICIT DispatchSequenceNumber}
```

The data type for a Submisssion time is defined as follows:

```
SubmissionTime ::= Time
```

The entries in the Outlog or the Auto-correlation Log may be subsetted by specifying a "range" or "cutoff" defined by Dispatch Sequence Number or Submission Time:

```
DispatchRange ::= SEQUENCE {
    from[0]DispatchPosition OPTIONAL --omitted means no lower bound--,
    to [1] DispatchPosition OPTIONAL --omitted means no upper bound--}

DispatchCutoff ::= SEQUENCE {
    to [1] DispatchPosition OPTIONAL --omitted means no upper bound--}

DispatchPosition ::= CHOICE {
    number [0] IMPLICIT DispatchSequenceNumber,
    time [1] IMPLICIT SubmissionTime}
```

10.2.2.4 Object Request

Several operations accept in the argument a request that an object (or objects) be returned in the result by reference rather than by value. A request for return by reference is used when the object is to be transferred to another "sink" service using the Third Party Transfer Service; if the Third Party Transfer service is not available, the object is returned by value instead of by reference.

The object request is represented by the following data type:

```
ObjRequest ::= [APPLICATION 29] INTEGER {byValue (0), byReference (1)}
```

10.2.2.5 Fetch Restrictions

The fetch operations can be made to filter by means of fetch restrictions. The Restrictions data type is defined as follows:

```
Restrictions ::= SET {
    allowedContentTypes SET OF ContentType OPTIONAL
        --default is no restriction--,
    allowedEITs EITs OPTIONAL --default is no restriction--,
    maximumContentLength INTEGER OPTIONAL --default is no restriction--}
```

The OverrideRestrictions data type, which allows one or more restrictions to be temporarily suppressed, is defined as follows:

```
OverrideRestrictions ::= BIT STRING {
    overrideContentTypesRestriction (0),
    overrideEITsRestriction (1),
    overrideContentLengthRestriction (2) }
```

10.2.2.6 Mailbox Attributes

The protocol recognizes a variety of Mailbox Attributes. The type of the Attribute consists of an object identifier:

```
AttributeType ::= OBJECT IDENTIFIER
```

The data type used for the value of an Attribute is defined for any particular Attribute type by means of the ATTRIBUTE (see Appendix B) macro defining that Attribute type. The data type so defined is used within the AttributeValue data type as follows:

```
AttributeValue ::= [APPLICATION 28] ANY
    -- the data type is defined by the Attribute type --
```

The Attributes in a Mailbox entry such as an Inlog or Outlog Entry are returned as a set of Attribute information, each represented by the following data type:

```
AttributeInformation ::= SEQUENCE {
    AttributeType,
    CHOICE {
        AttributeValue,--if Attribute is defined not to be multivalued --,
        SET OF AttributeValue ,
            -- if Attribute is defined to be multivalued--}}
```

10.2.2.7 Filters

The data type for a filter is defined as :

```
Filter:: = CHOICE {
    item [0] FilterItem,
    and [1] IMPLICIT SET OF SimpleFilter,
    or [2] IMPLICIT SET OF SimpleFilter,
    not [3] SimpleFilter -- simple filter must not use "not"--}

SimpleFilter:: = CHOICE {
    item [0] FilterItem,
    not [3] SimpleFilter }
```

When applied via an Operation to an Entry in a Mailbox Object, the filter results in a Boolean value (TRUE or FALSE) indicating whether or not that Entry should be part of the result of the operation.

The filter can take one of four forms. In each of these forms the filter consists of a number of terms.

In the first form the filter consists of only one term, and the filter becomes TRUE for an Entry if the term is TRUE for that Entry.

In the second form the filter consists of a set of terms and the filter becomes TRUE for an Entry if all the terms are TRUE for that Entry.

In the third form the filter consists of a set of terms and the filter becomes TRUE for an Entry if at least one of the terms is TRUE for the Entry.

In the fourth form the filter consists of one term and the filter becomes TRUE for an Entry if the term is FALSE for that Entry.

In the first and fourth form of the filter the term is a filter item; in the second and third form each of the terms of the filter is either a filter item or the logical inverse of a filter item.

This implies that in this version of the standard there is only one level of nesting of filter items in a filter.

The data type for a filter item is defined as :

```
FilterItem:: = CHOICE{
    equality [0] IMPLICIT AttributeValueAssertion
        --substrings [1] reserved for future use--,
    greaterOrEqual [2] IMPLICIT AttributeValueAssertion,
        --only for integers and times--,
    lessOrEqual [3] IMPLICIT AttributeValueAssertion,
        --only for integers and times--,
    present [4] IMPLICIT AttributeType}

AttributeValueAssertion::= SEQUENCE {
    AttributeType,
    AttributeValue }
```

A filter item can take one of five forms.

When any of these forms is used as part of the filter parameter of an operation applied to an Entry in a Mailbox Object, it results in a Boolean value. The way in which the Boolean value is derived from the filter item and the Entry depends on the form and is described in clause 8.5.

10.2.2.8 Selectors

The protocol frequently requires to select MS Messages or log entries by specifying a range of messages or entries to be considered and a filter of the messages or entries in that range, with limit on the total number of the messages or entries thus specified :

```
MSSelector :: = SET {
    range [0] IMPLICIT MSRange OPTIONAL --default is unbounded--,
    filter [1] Filter OPTIONAL
        -- default is all messages or entries in range--,
    limit [2] IMPLICIT INTEGER OPTIONAL -- default is no limit--}

DispatchSelector :: = SET {
    range [0] IMPLICIT DispatchRange OPTIONAL --default is unbounded--,
    filter [1] Filter OPTIONAL --default is all entries in range--,
    limit [2] IMPLICIT INTEGER OPTIONAL --default is no limit-- }
```

10.2.2.9 Return Classes

The protocol recognizes a variety of returns which are automatically correlated in the Auto-correlation Log against the original message. Each Return Class is denoted by an Object Identifier (see Appendix A):

```
ReturnClass      ::= OBJECT IDENTIFIER
```

The auto-correlation takes place against an auto-correlation element, whose data type is defined by each Return Class definition :

```
Auto CorrelationElement :: = ANY
```

10.2.3 Remote Operations

The protocol comprises remote operations in the following service groups of clause 9.2 :

Basic Mailbox
Dispatch
Message Store
Inlog
Outlog

Auto-correlation information
Registration (of Automatic Actions)
Status and inform

10.2.3.1 Basic Mailbox Operations

The protocol contains the following operations of a basic nature:

- Bind

- Unbind

Bind

The bind operation begins a session with the MBS. It accepts as argument the credentials of the initating user, the Mailbox Subset (General or IPM) required, the O/R name of the Mailbox, and any fetch restrictions to be in effect during the binding.

The MBS should accept the absence of security attributes from the MBC only in an environment with a zero security policy.

Unbind

The unbind operation ends a binding with the MBS.

10.2.3.2 Dispatch Operations

The protocol contains the following dispatch operations:

- Dispatch message

- Dispatch probe

- Cancel

Dispatch Message

The dispatchMessage operation dispatches a message of any content type to the MBS. The dispatched message is directly submitted to the MTSL (even if Deferred Delivery is requested). It accepts as argument the dispatch envelope and the content of the message. The envelope contains parameters controlling submission to the MTSL, together with an indication of whether the auto-correlation services are to be applied to it. The content may be supplied explicitly or by a reference for the third-party transfer service. It returns as result the message's Dispatch Sequence Number, assigned by the MBS, and the time of submission to the MTSL.

Dispatch Probe

The dispatchProbe operation dispatches a probe to the MBS. The dispatched probe is directly passed to the MTSL. It accepts as argument an envelope containing parameters controlling the probe request passed to the MTSL, together with an indication of whether the auto-correlation services are to be applied to it. It returns as result the probe's Dispatch Sequence Number, assigned by the MBS, and the time of "submitting" the probe request to the MTSL.

Cancel

The cancel operation retracts a previously made dispatch which requested deferred delivery. The MBS passes the request on to the MTSL, which may be unable to effect the cancellation. It accepts as argument the Dispatch Sequence Number of the message.

Note 6

The MBSE may have to translate the Dispatch Sequence Number to the Submit Event ID to perform the cancellation.

10.2.3.3 Message Store Operations

The protocol contains the following operations having to do with the Message Store:

- Analyse MS
- List messages
- Fetch message
- Delete messages

Analyse MS

The analyseMS operation returns analyses of messages in the MS. It accepts as argument a range of the MS Messages to be considered, a filter of the messages within that range, and, for each analysis requested, the Attribute type or the pair of Attributes types against which the analysis is to be performed. It returns as result a count of the messages considered, their lowest and highest MS Sequence Numbers, and the requested analyses.

List Messages

The listMessages operation returns MS Entries from one or more messages in the Message Store. It accepts as argument the range of messages to be considered, a filter of the messages within that range, the types of Attribute information to be returned from each MS Entry, and a limit on the number of entries to be returned. The list of MS Entries is itself an object, and this may be returned by value or as a third-party transfer reference. It returns as result the requested MS Entries. If there were MS Entries which could not be returned because of the limit imposed, the MS Sequence Number of the next entry is also returned.

Fetch Message

The fetchMessage operation retrieves an MS Message. It accepts as argument either the MS Sequence Number of the desired message or a search argument for the first message within a range of MS Messages; the messages within that range to be considered may be further reduced by specifying a filter. The elements to be returned from the MS Message may be selected; the selection parameter can distinguish between an MS Content containing a delivered content and a content returned with a Non-delivery Notification. The types of Attribute information to be returned from the MS Entry may be specified. The current fetch restrictions are applied when fetching the content (except if overriden). The content is an object, and this may be returned by value or as a third-party transfer reference.

It returns as result the selected elements from the MS Message. If the MS Message was specified by a search argument, a list of MS Sequence Numbers for other messages meeting the search criteria, up to the maximum limit, if such a limit was given and the MS Sequence Number of the next message if the limit was exceeded.

Delete Messages

The deleteMessages operation deletes messages from the MS. It accepts as argument either a range of the messages or a list of the MS Sequence Numbers of the messages to be deleted. No messages are deleted if any of the selected messages has an MS Status of "New".

10.2.3.4 Inlog Operations

The protocol contains the following operations having to do with the Inlog:

Analyse Inlog
List Inlog Entries
Delete Inlog Entries

Analyse Inlog

The analyse Inlog operation returns analyses of entries in the Inlog. It accepts as argument a range of the Inlog Entries to be considered, a filter of the entries within that range, and, for each analysis requested, the Attribute type or the pair of Attribute types against which the analysis is to be performed. It returns as result a count of the Inlog Entries considered, their lowest and highest MS Sequence Numbers, and the requested analyses.

List Inlog Entries

The listInlogEntries operation returns one or more Inlog Entries. It accepts as argument the range of entries are to be considered, a filter of the entries within that range, the types of Attribute information to be returned from each entry, and a limit on the number of entries to be returned. The list of entries is itself an object, and this may be returned by value or as a third-party transfer reference. It returns as result the requested Inlog Entries. If there were entries which could not be returned because of the limit imposed, the MS Sequence Number of the next entry is also returned.

Delete Inlog Entries

The deleteInlogEntries operation deletes the Inlog Entries up to and including the cutoff specified as argument.

10.2.3.5 Outlog Operations

The protocol contains the following operations having to do with the Outlog:

Analyse Outlog
List Outlog Entries
Delete Outlog Entries

Analyse Outlog

The analyse Outlog operation returns analyses of entries in the Outlog. It accepts as argument a range of the Outlog Entries to be considered, a filter of the entries within that range, and, for each analysis requested, the Attribute type or the pair of Attribute types against which the analysis is to be performed. It returns as result a count of the Outlog Entries considered, their lowest and highest Dispatch Sequence Numbers, and the requested analyses.

List Outlog Entries

The listOutlogEntries operation returns one or more Outlog Entries. It accepts as argument the range of entries are to be considered, a filter of the entries within that range, the types of Attribute information to be returned from each entry, and a limit on the number of entries to be returned. The list of entries is itself an object, and this may be returned by value or as a third-party transfer reference. It returns as result the requested Outlog Entries. If there were entries which could not be returned because of the limit imposed, the Dispatch Sequence Number of the next entry is also returned.

Delete Outlog Entries

The deleteOutlogEntries operation deletes the Outlog Entries up to and including the cutoff specified as argument.

10.2.3.6 Auto-correlation Information Operations

The protocol contains the following operations having to do with the Auto-correlation Log:

List AC Log Entries
Read AC Log Entry
Delete AC Log Entry

List AC Log Entries

The listACLogEntries operation returns synopses for selected Return Classes of one or more Auto-correlation Log Entries. It accepts as argument the range of entries are to be considered, and a limit on the number of synopses to be returned. The list of synopses is itself an object, and this may be returned by value or as a third-party transfer reference. It returns as result the requested synopses. If there were entries for which synopses could not be returned because of the limit imposed, the Dispatch Sequence Number of the next entry is also returned.

Read AC Log Entry

The readACLogEntry operation retrieves the information for selected Return Classes from a specified Auto-correlation Log Entry. It accepts as argument the Dispatch Sequence Number of the Entry to be returned as result. The AC Log Entry is itself an object, and this may be returned by value or as a third-party transfer reference.

Delete AC Log Entry

The deleteACLogEntry operation deletes the Auto-correlation Log Entry for the message whose Dispatch Sequence Number is supplied as argument. It also deletes the corresponding MS Entries for certain Return Classes.

10.2.3.7 Registration of Automatic Actions

The protocol contains the following operation for registration of automatic actions:

Register auto-actions

Register Auto-Actions

The registerAutoActions operation activates or deactivates the performing by the MBS of the automatic Actions specified in the argument. Note that it is permissible to deactivate an Action which had not been previously activated. It returns as result a list of all the automatic Actions currently registered.

10.2.3.8 Registration of Message Transfer Services

The protocol contains the following operation for registration with the MTA :

Register with MTA

Register with MTA

The register with MTA operation may register a new UA "profile" with the MTA, it may activate or deactivate the registration of recipient redirection for a period of time, and it may activate or deactivate the hold for Delivery Service. The profile concerns the messages that are deliverable to the MBS and consists of:

registered content type capability
registered encoded information types
registered content length capability

10.2.4 Remote Errors

The protocol contains remote errors. These relate to semantic errors in the usage of the operations by the MBC. Syntactic errors are a protocol violation and should be rejected.

The use of an operation or of an object identifier outside those defined for the Mailbox Subset agreed at the time of binding constitutes a syntactic error.

10.2.5 The P7Operations Module

This clause is a formal definition of the P7Operations module

```
P7Operations DEFINITIONS ::=
BEGIN
```

```
-- Imported from ECMA TR/31 --

        BIND ::= RemoteOperations.BIND
        UNBIND ::= RemoteOperations.UNBIND
        OPERATION ::= RemoteOperations.OPERATION
        ERROR ::= RemoteOperations.ERROR

-- Imported from ECMA TR/42 --

        TPT-DATA        ::= ECMA-TR/42.TPT-DATA

-- Imported from ECMA-93 --

        ContentType ::= P1.ContentType
        EITs ::= P1.EncodedInformationTypes
        ExplicitConversion ::= P1.ExplicitConversion
        ORName ::= P1.ORName
        PerMessageFlag ::= P1.PerMessageFlag
        Priority ::= P1.Priority
        ReassignmentInfo ::= P1.ReassignmentInfo
        Report ::= P1.Report
        SupplementaryInformation ::= P1.SupplementaryInformation
        Time ::= P1.Time
        UMPDUContent ::= P1.UMPDUContent

-- Imported from ECMA TR/32 --

        Name ::= DirectoryService.Name

-- Bind and unbind --

        bind    BIND
                ARGUMENT SET {
                    securityAttributes [0] IMPLICIT SET {
                        credentials [0] CHOICE{
                            [0] IMPLICIT SimpleCredentials } OPTIONAL },
                    mailboxSubset [1] CHOICE {
                        INTEGER {generalSubset (0), ipmSubset(1) } },
                    mailboxName [2] IMPLICIT ORName,
                    fetchRestrictions [3] IMPLICIT Restrictions OPTIONAL
                        --default is none-- }
                RESULT SET {
                        securityAttributes [0] IMPLICIT SET { }}
                BIND-ERRORS
                        MailboxBindError
                ::= 1


        unbind  UNBIND
                ::=2


-- Dispatch operations --

        dispatchMessage     OPERATION
                            ARGUMENT SEQUENCE {
                                envelope DispatchEnvelope,
                                content DispatchContent}
                            RESULT DispatchResult
```

```
                    ERRORS {
                        serviceNotAvailable,
                        accessControlViolation
                        mTANotAvailable,
                        invalidRedispatch,
                        thirdPartyError,
                        submitOrProbeError}
                    ::= 101

dispatchProbe   OPERATION
                ARGUMENT DispatchProbeEnvelope
                RESULT DispatchResult
                ERRORS {
                    serviceNotAvailable,
                    accessControlViolation,
                    mTANotAvailable,
                    submitOrProbeError}
                ::= 102

cancel  OPERATION
        ARGUMENT DispatchSequenceNumber
        RESULT NULL
        ERRORS {
            serviceNotAvailable,
            accessControlViolation,
            mTANotAvailable,
            invalidDispatchSequenceNumber,
            cancelError}
        ::= 103

-- Message Store Operations --

analyseMS  OPERATION
           ARGUMENT SET {
               selector [0] IMPLICIT MSSelector,
               analysisRequests [1] IMPLICIT SEQUENCE OF
                   AnalysisRequest OPTIONAL
                       --absent if no analyses are requested--}
           RESULT SET {
               next [0] IMPLICIT MSSequenceNumber OPTIONAL
                   --if there are more messages than the limit--,
               count [1] IMPLICIT INTEGER
                   --of the messages selected--,
               span [2] IMPLICIT MSSpan--of the messages selected--,
               analyses [3] IMPLICIT SEQUENCE OF Analysis OPTIONAL
                   --same order as the analysis requests-- }
           ERRORS {
               serviceNotAvailable,
               accessControlViolation,
               invalidRange,
               invalidFilter,
               invalidLimit,
               invalidAnalysisRequest,
               noEntries}
           ::= 104
```

```
listMessages    OPERATION
                ARGUMENT SET {
                    selector [0] IMPLICIT MSSelector,
                    requestedAttributes [1] IMPLICIT SET OF AttributeType
                        OPTIONAL
                            --default is all available Attribute types--,
                    ObjRequest OPTIONAL --default is by value -- }
                RESULT SET {
                    next [0] IMPLICIT MSSequenceNumber OPTIONAL
                        --if there are more messages than the limit--,
                    requested [1] TPT-DATA {MSList}
                        --by value or reference according to ObjRequest--}
                ERRORS {
                    serviceNotAvailable,
                    attributeNotAvailable,
                    accessControlViolation,
                    invalidRange,
                    invalidFilter,
                    invalidLimit,
                    noEntries}
                ::= 105

fetchMessage    OPERATION
                ARGUMENT FetchArgument
                RESULT SET {
                    MSMessage,
                    list[0]IMPLICIT SEQUENCE OF MSSequenceNumber
                        OPTIONAL,
                    next[1] IMPLICIT MSSequenceNumber OPTIONAL
                        --if there are more messages than the limit--}
                ERRORS {
                    serviceNotAvailable,
                    attributeNotAvailable,
                    accessControlViolation,
                    invalidMSSequenceNumber,
                    invalidRange,
                    invalidFilter,
                    invalidLimit,
                    noEntries,
                    fetchRestrictionViolation }
                ::= 106

deleteMessages      OPERATION
                    ARGUMENT SET {
                        messages [0] CHOICE {
                            range [0] IMPLICIT MSRange,
                            numbers [1] IMPLICIT SET OF
                                MSSequenceNumber}}
                    RESULT NULL
```

```
        ERRORS {
            serviceNotAvailable,
            accessControlViolation,
            invalidMSSequenceNumber,
            invalidRange,
            newMessagesPresent }
        ::= 107

-- Inlog operations --

    analyseInlog    OPERATION
                    ARGUMENT SET {
                        selector [0] IMPLICIT MSSelector,
                        analysisRequests [1] IMPLICIT SEQUENCE OF
                            AnalysisRequest OPTIONAL
                                -- absent if no analyses are requested--}
                    RESULT SET {
                        next [0] IMPLICIT MSSequenceNumber OPTIONAL
                            --if there are more entries than the limit--,
                        count [1] IMPLICIT INTEGER --of the entries selected--,
                        span [2] IMPLICIT MSSpan --of the entries selected--,
                        analyses [3] IMPLICIT SEQUENCE OF Analysis
                            OPTIONAL
                            --same order as the analysis requests-- }
                    ERRORS {
                        serviceNotAvailable,
                        accessControlViolation,
                        invalidRange,
                        invalidFilter,
                        invalidLimit,
                        invalidAnalysisRequest,
                        noEntries }
                    ::= 108

    listInlogEntries OPERATION
                    ARGUMENT SET {
                        selector [0] IMPLICIT MSSelector,
                        requestedAttributes [1] IMPLICIT SET OF AttributeType
                            OPTIONAL
                                --default is all available Attribute types--,
                        ObjRequest OPTIONAL --default is by value-- }
                    RESULT SET {
                        next [0] IMPLICIT MSSequenceNumber OPTIONAL
                            --if there are more entries than the limit--,
                        requested [1] TPT-DATA {InlogList }
                            --by value or reference according to ObjRequest--}
                    ERRORS {
                        serviceNotAvailable,
                        accessControlViolation,
                        invalidRange,
                        invalidLimit,
                        invalidFilter,
                        attributeNotAvailable,
                        noEntries }
                    ::= 109
```

```
deleteInlogEntries    OPERATION
                      ARGUMENT SET {
                          cutoff [0] IMPLICIT MSCutoff }
                      RESULT NULL
                      ERRORS {
                          serviceNotAvailable,
                          accessControlViolation,
                          noEntries,
                          mSmessagesNotYetDeleted}
                      ::= 110

-- Outlog operations --

analyseOutlog         OPERATION
                      ARGUMENT SET {
                          selector [0] IMPLICIT DispatchSelector,
                          analysisRequests [1] IMPLICIT SEQUENCE OF
                              AnalysisRequest OPTIONAL
                                  --absent if no analyses are requested--}
                      RESULT SET {
                          next [0] IMPLICIT DispatchSequenceNumber OPTIONAL
                              --if there are more entries than the limit--,
                          count [1] IMPLICIT INTEGER
                              --of the entries selected--,
                          span [2] IMPLICIT DispatchSpan
                              --of the entries selected--,
                          analyses [3] IMPLICIT SEQUENCE OF Analysis
                              OPTIONAL
                              --same order as the analysis requests-- }
                      ERRORS {
                          serviceNotAvailable,
                          accessControlViolation,
                          invalidRange,
                          invalidFilter,
                          invalidLimit,
                          invalidAnalysisRequest,
                          noEntries }
                      ::= 111

listOutlogEntries     OPERATION
                      ARGUMENT SET {
                          selector [0] IMPLICIT DispatchSelector,
                          requestedAttributes [1] IMPLICIT SET OF
                              AttributeType OPTIONAL
                                  --default is all available Attribute types--,
                          ObjRequest OPTIONAL --default is by value-- }
                      RESULT SET {
                          next [0] IMPLICIT DispatchSequenceNumber
                              OPTIONAL
                                  --if there are more entries than the limit--,
                          requested [1] TPT-DATA {OutlogList}
                              --by value or reference according to ObjRequest--}
```

```
                    ERRORS {
                        serviceNotAvailable,
                        accessControlViolation,
                        invalidRange,
                        invalidLimit,
                        invalidFilter,
                        attributeNotAvailable,
                        noEntries }
                    ::= 112

deleteOutlogEntries     OPERATION
                        ARGUMENT SET {
                            cutoff [0] IMPLICIT DispatchCutoff}
                        RESULT NULL
                        ERRORS {
                            serviceNotAvailable,
                            accessControlViolation,
                            noEntries}
                        ::= 113

-- Auto-correlation Log Operations --

listACLogEntries    OPERATION
                    ARGUMENT SET {
                        range [0] IMPLICIT DispatchRange OPTIONAL
                            --default is unbounded--,
                        limit [1] IMPLICIT INTEGER OPTIONAL
                            --default is no limit--,
                        requestedReturnClasses [2] IMPLICIT SET OF
                            ReturnClass OPTIONAL
                                --default is all classes for which--
                                --auto-correlation was requested--
                                --when the message was dispatched--,
                        ObjRequest OPTIONAL --default is by value-- }
                    RESULT SET {
                        next [0] IMPLICIT DispatchSequenceNumber
                            OPTIONAL
                                --if there are more entries than the limit--,
                        span [1] IMPLICIT DispatchSpan,
                        requested [2] TPT-DATA {ACLogList}
                            --by value or reference according to ObjRequest--
                                --one for each ReturnClass requested --}
                    ERRORS {
                        serviceNotAvailable,
                        returnClassNotAvailable,
                        accessControlViolation,
                        invalidRange,
                        invalidLimit,
                        noEntries}
                    ::= 114
```

```
readACLogEntry      OPERATION
                    ARGUMENT SET {
                        requestedEntry [0] IMPLICIT
                            DispatchSequenceNumber,
                        requestedReturnClasses [1] IMPLICIT SET OF
                            ReturnClass OPTIONAL
                                --default is all classes for which--
                                --auto-correlation was requested--
                                --when the message was dispatched--,
                        ObjRequest OPTIONAL--default is by value--}
                    RESULT SET {
                        requested  TPT-DATA {ACLogEntry}
                    ERRORS {
                        serviceNotAvailable,
                        returnClassNotAvailable,
                        accessControlViolation,
                        invalidDispatchSequenceNumber,
                        returnClassNotRequested }
                    ::= 115

deleteACLogEntry    OPERATION
                    ARGUMENT SET {
                        requestedEntry [0] IMPLICIT
                            DispatchSequenceNumber}
                    RESULT NULL
                    ERRORS {
                        serviceNotAvailable,
                        accessControlViolation,
                        invalidDispatchSequenceNumber}
                    ::= 116

-- Registration of automatic actions --

registerAutoActions     OPERATION
                        ARGUMENT SET OF AutoActionRequest
                        RESULT SET OF AutoAction--registered
                        ERRORS {
                            serviceNotAvailable,
                            accessControlViolation,
                            invalidAutoActionRequest}
                        ::= 117

-- Register with MTA --

registerWithMTA     OPERATION
                    ARGUMENT SET {
                        --if a member of this set is absent--
                        --there is no change in the corresponding registration--
                            newProfile [0] IMPLICIT UAProfile OPTIONAL,
                            changeRecipientRedirection [1] IMPLICIT
                                RecipientRedirectionRequest OPTIONAL,
                            changeHoldForDelivery [2]
                                IMPLICIT HoldForDeliveryRequest
                                    OPTIONAL}
                    RESULT SET { }
```

```
        ERRORS {
            serviceNotAvailable,
            accessControlViolation,
            mTANotAvailable,
            invalidUAProfile,
            invalidRecipientRedirectionRequest,
            holdForDeliveryError }
        ::= 118

-- Common data types --

    MSStatus::= INTEGER {
        new (0), listed (1), processed (2), deleted (3)}

-- MS Sequence Number and Delivery Time --

    MSSequenceNumber::= INTEGER


    MSSpan::= SEQUENCE {
        lowest [0] IMPLICIT MSSequenceNumber,
        highest [1] IMPLICIT MSSequenceNumber}

    DeliveryTime::= Time

    MSRange::= SEQUENCE {
        from [0] MSPosition OPTIONAL --omitted means no lower bound--,
        to [1] MSPosition OPTIONAL --omitted means no upper bound--}

    MSCutoff::= SEQUENCE {
        to [1] MSPosition OPTIONAL --omitted means no upper bound--}

    MSPosition::= CHOICE {
        number [0] IMPLICIT MSSequenceNumber,
        time [1] IMPLICIT DeliveryTime }

-- Dispatch Sequence Number and Submission Time --

    DispatchSequenceNumber::= INTEGER

    DispatchSpan::= SEQUENCE {
        lowest [0] IMPLICIT DispatchSequenceNumber,
        highest [1] IMPLICIT DispatchSequenceNumber }

    Submission Time::= Time

    DispatchRange::= SEQUENCE {
        from [0] DispatchPosition OPTIONAL --omitted means no lower bound--,
        to [1] DispatchPosition OPTIONAL --omitted means no upper bound--}

    DispatchCutoff::= SEQUENCE {
        to [1] DispatchPosition OPTIONAL --omitted means no upper bound--}

    DispatchPosition::= CHOICE {
        number [0] IMPLICIT DispatchSequenceNumber,
        time [1] IMPLICIT SubmissionTime}

-- Third Party Transfer Object request --

    ObjRequest::= [APPLICATION 29] INTEGER {byValue (0),byReference (1)}
```

```
-- Fetch restrictions --

        Restrictions::= SET {
            allowedContentTypes SET OF ContentType OPTIONAL
                --default is no restriction--,
            allowedEITs EITs OPTIONAL --default is no restriction--,
            maximumContentLength INTEGER OPTIONAL --default is no restriction--}

        OverrideRestrictions::= BIT STRING {
            overrideContentTypesRestriction (0),
            overrideEITsRestriction (1),
            overrideContentLengthRestriction (2) }

-- Attributes --

        AttributeType::= OBJECT IDENTIFIER


        AttributeValue::= [APPLICATION 28] ANY
            --the data type is defined by the Attribute type --


        AttributeInformation::= SEQUENCE {
            AttributeType,
            CHOICE {
                AttributeValue--if Attribute is defined not to be multivalued--,
                SET OF AttributeValue--if Attribute is defined to be multivalued--}}

-- Filters --

        Filter::= CHOICE {
            item [0] FilterItem,
            and [1] IMPLICIT SET OF SimpleFilter,
            or [2] IMPLICIT SET OF SimpleFilter,
            not [3] SimpleFilter
                --the simple filter must not use "not"--}


        SimpleFilter::= CHOICE {
            item [0] FilterItem,
            not [3] SimpleFilter }


        FilterItem::= CHOICE {
            equality [0] IMPLICIT AttributeValueAssertion
                --substrings [1] reserved for future use--,
            greaterOrEqual [2] IMPLICIT AttributeValueAssertion
                --only for integers and times--,
            lessOrEqual [3] IMPLICIT AttributeValueAssertion
                --only for integers and times--,
            present [4] IMPLICIT AttributeType }


        AttributeValueAssertion::= SEQUENCE {
            AttributeType,
            AttributeValue }
```

```
-- Selectors --

        MSSelector::= SET {
            range [0] IMPLICIT MSRange OPTIONAL --default is unbounded--,
            filter [1] Filter OPTIONAL
                --default is all messages or entries in range--,
            limit [2] IMPLICIT INTEGER OPTIONAL --default is no limit-- }

        DispatchSelector::= SET {
            range [0] IMPLICIT DispatchRange OPTIONAL --default is unbounded--,
            filter [1] Filter OPTIONAL --default is all entries in range--,
            limit [2] IMPLICIT INTEGER OPTIONAL --default is no limit-- }

-- Return Classes --

        ReturnClass::= OBJECT IDENTIFIER


        AutoCorrelationElement::= ANY

-- Data types for binding --

        SimpleCredentials::= SEQUENCE {UserName, UserPassword}

        UserName::= Name

        UserPassword::= OCTET STRING

-- Dispatch --

        DispatchEnvelope::= SET {
            COMPONENTS OF SubmitParameters
                --EITs omitted implies that they are unspecified--,
            [29] IMPLICIT AutoCorrelationRequest OPTIONAL
                --default is no auto-correlation-- }

        SubmitParameters::= SET {
            originator ORName OPTIONAL--default is the name of the Mailbox--,
            EITs OPTIONAL,
            ContentType,
            deferredDelivery [0] IMPLICIT Time OPTIONAL,
            Priority DEFAULT normal,
            PerMessageFlag DEFAULT { },
            [1] IMPLICIT SEQUENCE OF SubmitRecipientInfo,
            latestDelivery [2] IMPLICIT Time OPTIONAL }

        SubmitRecipientInfo::= SET {
            recipient ORName,
            [0] IMPLICIT UserReportRequest DEFAULT {basic},
            [1] IMPLICIT ExplicitConversion OPTIONAL,
            originatorRequestedAlternativeRecipient [2] IMPLICIT ORName
                OPTIONAL}

        UserReportRequest::= BIT STRING {
            confirmed(3), basic(4) }

        AutoCorrelationRequest::= SET OF ReturnClass
            --one for each class of auto-correlation requested--

        DispatchContent::= CHOICE {
            content [0] TPT-DATA {UMPDUContent},
                --the representation of the objects data type is constrained by the- --
                    --content type--,
```

```
            redispatch [1] IMPLICIT MSSequenceNumber
                --of non-delivery notification whose returned content is to be--
                    --dispatched--}

        DispatchResult::= SET {
            submissionTime [0] IMPLICIT SubmissionTime,
            sequenceNumber [1] IMPLICIT DispatchSequenceNumber }

-- Probe --

        DispatchProbeEnvelope::= SET {
            COMPONENTS OF ProbeParameters
                --EITs omitted implies that they are unspecified--,
            [29] IMPLICIT AutoCorrelationRequest OPTIONAL
                --default is no auto-correlation-- }

        ProbeParameters::= SET {
            originator ORName OPTIONAL--default is the name of the Mailbox--,
            EITs OPTIONAL,
            ContentType,
            contentLength [0] IMPLICIT INTEGER OPTIONAL,
            PerMessageFlag DEFAULT { },
            [3] IMPLICIT SEQUENCE OF SubmitRecipientInfo}

-- Analysis --

        AnalysisRequest::= CHOICE {
            list AttributeType --single-valued--,
            table SEQUENCE {
                outer AttributeType --single-valued--,
                inner AttributeType --single-valued--} }

        Analysis::= CHOICE {
            list [0] IMPLICIT BasicAnalysis,
            table [1] IMPLICIT TableAnalysis }

        BasicAnalysis::= SET {
            absent [0] IMPLICIT INTEGER OPTIONAL
                --count of messages for which the attribute is absent--,
            present [1] IMPLICIT SET OF --one for each Attribute value present--
                SEQUENCE {
                    value AttributeValue,
                    count INTEGER } OPTIONAL }

        TableAnalysis::= SET {
            absent [0] IMPLICIT BasicAnalysis OPTIONAL
                --messages for which the outer attribute is absent,
                --further analysed by the inner attribute value--,
            present [1] IMPLICIT SET OF
                --one for each outer attribute value present--,
                SEQUENCE {
                    value AttributeValue --of the outer attribute--,
                    innerAnalysis BasicAnalysis} OPTIONAL }

-- Message Store --

        MSList::= SEQUENCE OF MSEntry

        MSEntry::= SET OF AttributeInformation
            --restricted to Attributes that may occur in an MS Entry--
```

```
FetchArgument::= SET {
    message [0] CHOICE {
        precise [0] IMPLICT MSSequenceNumber,
        search [1] IMPLICIT MSSelector},
    requestedElements [1] IMPLICIT MessageElements DEFAULT
        {envelope, deliveredContent},
    override [2] IMPLICIT OverrideRestrictions OPTIONAL
        --default is that any fetch restrictions in force do apply--,
    requestedAttributes [3] IMPLICIT SET OF AttributeType OPTIONAL
        --only relevant if entry is a requested element--
        --default is all available attribute types--,
    ObjRequest OPTIONAL --default is by value-- }

MessageElements::= BIT STRING {
    entry(0), envelope(1), deliveredContent(2), returnedContent(3)}

MSMessage::= SEQUENCE {
    entry [0] IMPLICIT MSEntry OPTIONAL--absent if entry not requested--
        --MS Status, if returned, is that prior to the fetch--,
    envelope [1] MSEnvelope OPTIONAL--absent if envelope not requested--,
    content [2] MSContent OPTIONAL
        --absent if neither delivered content nor returned content--
        --requested--
        --or (for a non-delivery report) if no content has been returned--}

MSEnvelope::= CHOICE {
    userMessage [0] IMPLICIT SET {
        deliveryTime [0] IMPLICIT DeliveryTime,
        deliverEnvelope [1] IMPLICIT DeliverEnvelope},
    deliveryReport [1] IMPLICIT SET {
        notificationTime [0] IMPLICIT DeliveryTime,
        dispatchSequenceNumber [1] IMPLICIT DispatchSequenceNumber
            --of original message--,
        notifications [2] IMPLICIT SEQUENCE OF NotifyRecipientInfo }}

DeliverEnvelope::= SET {
    [0] IMPLICIT ContentType,
    originator ORName,
    original [1] IMPLICIT EITs OPTIONAL,
    Priority DEFAULT normal,
    [2] IMPLICIT DeliveryFlags,
    otherRecipients [3] IMPLICIT SEQUENCE OF ORName OPTIONAL,
    thisRecipient [4] IMPLICIT ORName,
    intendedRecipient [5] IMPLICIT ORName OPTIONAL,
    converted [6] IMPLICIT EITs OPTIONAL,
    submission [7] IMPLICIT SubmissionTime,
    [9] IMPLICIT ReassignmentInfo OPTIONAL }

DeliveryFlags::= BIT STRING {conversionProhibited (1)}

NotifyRecipientInfo::= SET {
    recipient [0] IMPLICIT ORName,
    [1] Report,
    converted EITs OPTIONAL,
    intendedRecipient [2] IMPLICIT ORName OPTIONAL,
    [3] IMPLICIT SupplementaryInformation OPTIONAL }

MSContent::= TPT-DATA {UMPDUContent}
```

```
-- Inlog --

        InlogList::= SEQUENCE OF InlogEntry

        InlogEntry::= SET OF AttributeInformation
            --restricted to Attributes that may occur in an Inlog Entry--

-- Outlog --

        OutlogList::= SEQUENCE OF OutlogEntry

        OutlogEntry::= SET OF AttributeInformation
            --restricted to Attributes that may occur in an Outlog Entry--

-- Auto-correlation Log --

        ACLogList::= SEQUENCE OF ACLogEntrySynopsis

        ACLogEntrySynopsis::= SET {
            sequenceNumber [0] IMPLICIT DispatchSequenceNumber,
            time [1] IMPLICIT SubmissionTime,
            returnSynopses [2] IMPLICIT SET OF PerReturnClassSynopsis
                --one for each ReturnClass requested--}

        PerReturnClassSynopsis::= SEQUENCE {
            returnClass [0] IMPLICIT ReturnClass,
            [1] IMPLICIT SET OF PerReturnStatusInfo
                --one for each with non-zero count-- }

        PerReturnStatusInfo::= SEQUENCE {
            returnStatus [0] IMPLICIT ReturnStatus,
            count [1] IMPLICIT INTEGER }

        ReturnStatus::= INTEGER {
            negativeReturnNotReceived (0),
            returnNotReceived (1),
            negativeReturnReceived (2),
            positiveReturnInferred (3),
            positiveReturnReceived (4),
            returnReceived (5),
            unexpectedReturnReceived (6) }

        ACLogEntry::= SET {
            sequenceNumber [0] IMPLICIT DispatchSequenceNumber,
            time [1] IMPLICIT SubmissionTime,
            returns [2] IMPLICIT SET OF PerReturnClassInfo
                --one for each ReturnClass requested-- }

        PerReturnClassInfo::= SEQUENCE {
            returnClass [0] IMPLICIT ReturnClass,
            [1] IMPLICIT SEQUENCE OF DetailedReturn }

        DetailedReturn::= SEQUENCE {
            detailFor [0] AutoCorrelationElement,
            returnStatus [1] IMPLICIT ReturnStatus,
            associatedMSMessage [2] IMPLICIT MSSequenceNumber OPTIONAL
                --where relevant--}
```

```
-- Automatic actions --

        AutoActionRequest::= SET {
            requested [0] IMPLICIT ActionType
                --only allowed for Actions defined to be automatic--,
            activate [1] IMPLICIT BOOLEAN DEFAULT TRUE,
            registrationParameter [2] ANY OPTIONAL
                --of the type defined by the ACTION--
                    --omitted if activate is false-- }

        AutoAction::= SET {
            registered [0] IMPLICIT ActionType
                --defined as AUTOMATIC in an ACTION--,
            registrationParameter [2] ANY--of the type defined by the ACTION-- }

        ActionType::= OBJECT IDENTIFIER

-- Registration with MTA --

        UAProfile::= Restrictions

        RecipientRedirectionRequest::= SET {
            activate [0] IMPLICIT BOOLEAN
            redirectedTo [1] IMPLICIT ORName OPTIONAL
                --omitted if activate is false--
            expiry [2] IMPLICIT Time OPTIONAL --omitted if activate is false-- }

        HoldForDeliveryRequest::= SET {
            activate [0] IMPLICIT BOOLEAN }

-- Errors --

        MailboxBindError::= SET {
            refuseReason [0] IMPLICIT INTEGER {
                busy (0),
                securityFailure (1),
                invalidSubset (2),
                unrecognizedMailboxName (3)}}


        serviceNotAvailable ERROR
            PARAMETER SET {reason [0] IMPLICIT INTEGER {
                notSupported (0), notSubscribed (1), accessControlViolation(2)}}
            ::= 101

        attributeNotAvailable ERROR
            PARAMETER SET {reason [0] IMPLICIT INTEGER {
                notSupported (0), notSubscribed (1), accessControlViolation (2)}}
            ::= 102

        returnClassNotAvailable ERROR
            PARAMETER SET {reason [0] IMPLICIT INTEGER {
                notSupported (0), notSubscribed (1), accessControlViolation (2)}}
            ::= 103

        accessControlViolation ERROR
            PARAMETER SET {
                reason [0] IMPLICIT INTEGER {--other reasons for further study--
                    tooSensitive (0) } OPTIONAL }
            ::= 104

        mTANotAvailable ERROR
            PARAMETER SET {}
            ::= 105
```

```
invalidRedispatch ERROR
    PARAMETER SET {reason [0] IMPLICIT INTEGER {
        invalidMSSequenceNumber (0), wrongContentType (1) }
    ::= 106

thirdPartyError ERROR
    PARAMETER SET {}
    ::= 107

submitOrProbeError ERROR
    PARAMETER SET {
        reason [0] IMPLICIT INTEGER { -- given by MTSL--
            controlViolation (1),
            originatorInvalid (2),
            recipientImproperlySpecified (3),
            serviceElementNotSubscribed (4)},
        SEQUENCE OF ORName OPTIONAL
            --only if recipient improperly specified--}
    ::= 108

cancelError ERROR
    PARAMETER SET {
        reason [0] IMPLICIT INTEGER { --given by MTSL--
            invalidEventID (7),
            messageDelivered (8),
            messageTransferred (9)}}
    ::= 109

invalidMSSequenceNumber ERROR
    PARAMETER SET {}
    ::= 110

invalidDispatchSequenceNumber ERROR
    PARAMETER SET {}
    ::= 111

invalidRange ERROR
    PARAMETER SET {}
    ::= 112

invalidFilter ERROR
    PARAMETER SET {}
    ::= 113

invalidLimit ERROR
    PARAMETER SET {}
    ::= 114

newMessagePresent ERROR
    PARAMETER SET {}
    ::= 115

noEntries ERROR
    PARAMETER SET {}
    ::= 116

invalidAnalysisRequest ERROR
    PARAMETER SET {SET OF AnalysisRequest
        --not necessarily a complete list--}
    ::= 117
```

```
fetchRestrictionViolation ERROR
    PARAMETER SET {}
    ::= 118

msMessageNotYetDeleted ERROR
    PARAMETER SET {SET OF MSSequenceNumber
        --not necessarily a complete list--}
    ::= 119

returnClassNotRequested ERROR
    PARAMETER SET {SET OF ReturnClass--not necessarily a complete list--}
    ::= 120

invalidAutoActionRequest ERROR
    PARAMETER SET {SET OF AutoActionRequest
        --not necessarily a complete list--}
    ::= 121

invalidUAProfile ERROR
    PARAMETER SET {}
    ::= 122

invalidRecipientRedirectionRequest ERROR
    PARAMETER SET {}
    ::= 123

holdForDeliveryError ERROR
    PARAMETER SET {}
    ::= 124

END -- of module P7 Operations --
```

10.2.6 Object Identifiers

The protocol is potentially open-ended along a number of important dimensions, specifically the following:

- The Mailbox Attributes of the messages the MBS recognizes;

- the Actions the MBS will take automatically and/or log to the Action History Attribute;

- the Return Classes the MBS will use for auto-correlation.

Corresponding to each of the above dimensions is a set of Object Identifiers whose values denote specific types of Attributes, and so forth. The use of Object Identifiers facilitates future standard and non-standard extension of the protocol along the indicated lines.

The value of the base object identifier component "ecma-xxx" is to be defined.

10.2.6.1 Attribute Types

The object identifiers for Attribute types are defined by means of the ATTRIBUTE macro (see ECMA TR/42). The associated data type used for the Attribute value (or values if the Attribute is multivalued) is also defined.

The Attributes defined in clause 10.2.7 correspond to the General Mailbox Attributes of Tables 1 and 2.

The object identifier component attribute is defined by:

attribute OBJECT IDENTIFIER ::= {ecma-xxx 1}

10.2.6.2 Action Types

The protocol recognizes a variety of Actions involving messages delivered or notified to the MBS by the MTSL. These are defined using the ACTION macro of Appendix A.

The Actions defined in clause 10.2.7 correspond to the General Actions of Table 7.

The object identifier component action is defined as follows:

action OBJECT IDENTIFIER ::= {ecma-xxx 2}

10.2.6.3 Return Classes

The protocol recognizes a variety of Return Classes which are automatically correlated in the Auto-correlation Log against the original message.

The Return Class defined in clause 10.2.7 corresponds to the Auto-Correlation Request service described in clause 9.2.6.

The object identifier component returnClass is defined by:

returnClass OBJECT IDENTIFIER ::= {ecma-xxx 3}

10.2.7 The P7Identifiers Module

This clause is a formal definition of the P7Identifiers module.

```
P7Identifiers DEFINITIONS::=
BEGIN

-- Imported from P7Operations (clause 10.2.5) --

ActionType ::= P7Operations.ActionType
Content Type ::= P7Operations.ContentType
DeliveryTime ::= P7Operations.DeliveryTime
DispatchSequenceNumber ::= P7Operations.DispatchSequenceNumber
EITs ::= P7Operations.EITs
MSSequenceNumber ::= P7Operations.MSSequenceNumber
MSStatus ::= P7Operations.MSStatus
ORName ::= P7Operations.ORName
PerMessageFlag ::= P7Operations.PerMessageFlag
Priority ::= P7Operations.Priority
ReassignmentInfo ::= P7Operations.ReassignmentInfo
ReturnClass ::= P7Operations.ReturnClass
SubmissionTime ::= P7Operations.SubmissionTime
SubmitRecipientInfo ::= P7Operations.SubmitRecipientInfo
Time ::= P7Operations.Time
UserName ::= P7Operations.UserName

-- Imported from ECMA TR/42 --

ATTRIBUTE ::= ECMA-TR/42.ATTRIBUTE

-- Macros --

ACTION MACRO::=
BEGIN
    TYPE NOTATION ::= Registration Action | Action
    VALUE NOTATION::= value (VALUE OBJECT IDENTIFIER)
    Registration ::= "AUTOMATIC WITH REGISTRATION PARAMETER IS"
        type
    Action ::= "ACTION PARAMETER IS" type
END

RETURN-CLASS MACRO ::=
BEGIN
    TYPE NOTATION ::= "AUTO-CORRELATION ON"
        AutoCorrelationElement Qualifier
    VALUE NOTATION ::= value (VALUE OBJECT IDENTIFIER)
    AutoCorrelationElement ::= type
    Qualifier ::= "CONDITIONAL" | empty
END
```

```
-- Base object identifier --

        ecma-xxx OBJECT IDENTIFIER ::= {        }

-- Attributes --

        attribute OBJECT IDENTIFIER ::= {ecma-xxx 1}

        mSSequenceNumber ATTRIBUTE
            REPRESENTED BY MSSequenceNumber
            ::= {attribute 1}

        dispatchSequenceNumber ATTRIBUTE
            REPRESENTED BY DispatchSequenceNumber
        ::= {attribute 2}

        mBUser ATTRIBUTE
            REPRESENTED BY UserName
            ::= {attribute 3}

        mBAutoCorrelationRequest ATTRIBUTE
            MULTIVALUED
            REPRESENTED BY ReturnClass
            ::= {attribute 4}

        mSStatus ATTRIBUTE
            REPRESENTED BY MSStatus
            ::= {attribute 5}

        mSAutoCorrelated ATTRIBUTE
            REPRESENTED BY BOOLEAN
            ::= {attribute 6}

        mSActionHistory ATTRIBUTE
            MULTIVALUED
            REPRESENTED BY ActionRecord
            ::= {attribute 7}

        ActionRecord ::= SEQUENCE {
            time [0] IMPLICIT Time,
            action [1] IMPLICIT ActionType,
            initiator [2] UserName OPTIONAL
                --only if not automatic action--,
            actionParameter [3] ANY
                --type is defined by the ACTION macro--,
            registrationParameter [4] ANY OPTIONAL
                --type is defined by the ACTION macro--
                --present for automatic actions--}

        mBDispatchType ATTRIBUTE
            REPRESENTED BY DispatchType
            ::= {attribute 8}

        DispatchType ::= INTEGER {probe (0), message (1)}

        msDeliveryType ATTRIBUTE
            REPRESENTED BY DeliveryType
            ::= {attribute 9}

        DeliveryType ::= INTEGER {deliveredMessage (0), notification (1) }

        p1ContentType ATTRIBUTE
            REPRESENTED BY ContentType
            ::= {attribute 100}
```

```
p1Originator ATTRIBUTE
    REPRESENTED BY ORName
    ::= {attribute 101}

p1IntendedRecipient ATTRIBUTE
    REPRESENTED BY ORName
    ::= {attribute 102}

p1SubmissionTime ATTRIBUTE
    REPRESENTED BY SubmissionTime
    ::= {attribute 103}

p1DeliveryTime ATTRIBUTE
    REPRESENTED BY DeliveryTime
    ::= {attribute 104}

p1Priority  ATTRIBUTE
    REPRESENTED BY Priority
    ::= {attribute 105}

p1OriginalEITs ATTRIBUTE
    REPRESENTED BY EITs
    ::= {attribute 106}

p1DeliveredEITs ATTRIBUTE
    REPRESENTED BY EITs
    ::= {attribute 107}

p1ContentLength ATTRIBUTE
    REPRESENTED BY INTEGER
    ::= {attribute 108}

p1ThisRecipient ATTRIBUTE
    REPRESENTED BY ORName
    ::= {attribute 109}

p1OtherRecipient ATTRIBUTE
    MULTIVALUED
    REPRESENTED BY ORName
    ::= {attribute 110}

p1ContentReturned ATTRIBUTE
    REPRESENTED BY BOOLEAN
    ::= {attribute 111}

p1Converted ATTRIBUTE
    REPRESENTED BY BOOLEAN
    ::= {attribute 112}

p1ReassignmentInfo ATTRIBUTE
    MULTIVALUED
    REPRESENTED BY ReassignmentInfo
    ::= {attribute 113}

p1Cancelled ATTRIBUTE
    REPRESENTED BY BOOLEAN
    ::= {attribute 114}

p1DeferredDeliveryTime ATTRIBUTE
    REPRESENTED BY Time
    ::= {attribute 115}
```

```
        p1LatestDeliveryTime ATTRIBUTE
            REPRESENTED BY Time
            ::= {attribute 116}

        p1PerMessageFlag ATTRIBUTE
            REPRESENTED BY PerMessageFlag
            ::= {attribute 117}

        p1RecipientInfo ATTRIBUTE
            MULTIVALUED
            REPRESENTED BY SubmitRecipientInfo
            ::= {attribute 118}

        p1Recipient ATTRIBUTE
            MULTIVALUED
            REPRESENTED BY ORName
            ::= {attribute 119}

    -- Action types --

        action OBJECT IDENTIFIER::= {ecma-xxx 2}

        atFetched ACTION
            ACTION PARAMETER IS MSStatus
            ::= {action 1}

        atDeleted ACTION
            ACTION PARAMETER IS NULL
            ::= {action 2}

        atRedispatch ACTION
            ACTION PARAMETER IS DispatchSequenceNumber
            ::= {action 3}

    -- Return Classes --

        returnClass OBJECT IDENTIFIER::= {ecma-xxx 3}

        rcDeliveryNotification RETURN-CLASS
            AUTO-CORRELATION ON ORName --of the recipients--
            CONDITIONAL --upon User Report Request--
            ::= {returnClass 1}

    END --of P7Identifiers module --
```

10.3 Protocol Extensions Specific to Interpersonal Messaging

10.3.1 Introduction

The extensions to the general Mailbox Access Protocol that are specific to Interpersonal Messaging are defined in the following ASN.1 modules:

- P7ipmOperations: contains the definitions of the remote operations, the associated remote errors and the supporting data types (described in clauses 10.3.2 to 10.3.3 and summarized in clause 10.3.4):

- P7ipmIdentifiers: contains the definitions of the object identifiers used in conjunction with mailbox access for interpersonal messaging (described in clause 10.3.5 and summarized in clause 10.3.6):

The Mailbox Access Protocol incorporating these extensions is defined to contain the operations from both P7Operations and P7ipmOperations and the object identifiers from both P7Identifiers and P7ipmIdentifiers.

10.3.2 Remote Operations

10.3.2.1 Dispatch IP Message

The dispatchIPMessage operation builds an IP-message and submits it to the MTSL. The IP-message may be a reply to an IP-message in the MS; the reply may be to the originator of that message or to the list of users referenced in the replyToUsers element of that message. The complete content or the body parts of the message to be submitted may be supplied explicitly or by a reference for the third-party transfer service. IP-messages in the MS may be forwarded as body parts by specifying their MS Sequence Numbers.

The operation accepts as argument an indication of whether the Auto-correlation services are to be applied to it, along with an envelope, a heading, and a body consisting of a sequence of either body parts or the MS Sequence Numbers of IP-messages in the MS that are to be forwarded. If the IP-message is a reply, the argument also includes the MS Sequence Number of the IP-message with which the reply is associated and an indication of the type of reply.

The heading may be "incomplete" in the sense that the IPMessageId and originator elements may be omitted. They will instead be generated and filled in by the MBS.

The reply type is a choice between one or more of a reply to the originator, a reply to the list of users referenced in the replyToUsers element of the message being replied to, and a reply to all the original recipients. These recipients are treated as primary recipients. Additional recipients (primary, copy and blind copy) may be specified. Such additional primary recipients are placed after those derived from the original heading.

If the reply is to the originator, the O/R descriptor generated as a primary recipient in the heading is taken from the originator in the heading of the original message. If this element is absent or does not contain an O/R name, the O/R name taken from the delivery parameters is placed in the generated O/R descriptor.

If the reply is to the replyToUsers, each O/R descriptor will already contain an O/R name; if the replyToUsers element contains an O/R descriptor for this Mailbox, that O/R descriptor is ignored.

If the reply is to all the original recipients, the O/R descriptor for this Mailbox is ignored.

Several elements of the heading of the submitted IP-message will be derived by the MBS from the heading of the IP-message being replied to if they are not present in the heading presented in the argument. This applies to:

inReplyTo

subject

importance

sensitivity

The envelope may be "incomplete" in the sense that the MBS can be left to determine and fill in the EITs component. (Note that the MBS is unable to do this if the body consists of a single TTX or G3Fax body part and the associated non-basic parameters have been omitted from that body part; in this case the operation returns an error).

The MBS can be directed to extract the list of recipients used for submission from the heading, in which case the relevant O/R descriptors must contain an O/R name. In the case of a reply, the new heading is constructed before this extraction takes place. The list can be one of:

- all primary and copy recipients (in which case there must be no blind copy recipients);

- all blind copy recipients.

The operation returns as result the message's Dispatch Sequence Number, assigned by the MBS, the message's IP-message Id, and the Submission Time to the MTSL.

10.3.2.2 Issue IPM Notification

The issueIPMNotification operation generates a receipt or a non-receipt notification for a previously delivered IP-message as an IPM-status-report and submits it to the MTSL. It accepts as argument the MS Sequence Number of the message, and, for non-receipt, the reason for non-receipt. If the reason indicates uAEInitiatedDiscard, a non-receipt qualifier is also given, indicating expired or obsoleted. The non-receipt notification will only contain returned content if the originator requested its return.The argument also includes an indication of whether the Auto-correlation of Delivery and Non-delivery Notifications service is to be applied to it.

The operation returns as result the message's Dispatch Sequence Number, assigned by the MBS, the message's IP-message Id, and the Submission Time to the MTSL.

Note 7

The recipient for receipt and non-receipt notifications is derived in the same way as for the reply to IP-message (see clause 10.3.2.1).

10.3.2.3 IPM Fetch Message

The iPMFetchMessage operation retrieves from the MS either an interpersonal message (either an IP-message or an IPM-status-report) or a delivery/non-delivery notification for such a message. It is the same as the fetchMessage operation of the General Subset (see clause 10.2.3.3, Fetch Message) except for some additional functionality.

A part selector parameter if present allows selected parts (heading and body parts at any level of the nesting) of an IP-message in the MS Content to be extracted. Each part is itself an object, and may be returned by value or as a third-party transfer reference. If the message is an IPM-status-report, this breakdown applies to the returned content, if any.

If the parts selector parameter is absent, the content is returned in the same manner as for fetchMessage, except that the returned content with an IPM-status-report is suppressed if "returned content" is not requested.

10.3.3 Remote Errors

The protocol contains remote errors. These relate to semantic errors in the usage of the operations by the MBC.

10.3.4 The P7ipmOperations Module

This clause is a formal definition of the P7ipmOperations module.

```
P7ipmOperations DEFINITIONS::=
BEGIN

-- Imported from ECMA TR/31 --

    OPERATION ::= RemoteOperations.OPERATION
    ERROR ::= RemoteOperations.ERROR

-- Imported from ECMA TR/42 --

    TPT-DATA  ::= ECMA TR/42.TPT-DATA
```

```
-- Imported from ECMA-93 --

        BodyPart ::= P2.BodyPart
        Heading ::= P2.Heading
        Importance ::= P2.Importance
        IPMessageId ::= P2.IPMessageId
        NonReceiptReason ::= P2.NonReceiptReason
        NonReceiptQualifier ::= P2.NonReceiptQualifier
        ObsoletingTime ::= P2.ObsoletingTime
        ORDescriptor ::= P2.ORDescriptor
        Recipient ::= P2.Recipient
        ReportRequest ::= P2.ReportRequest
        Sensitivity ::= P2.Sensitivity
        Subject ::= P2.Subject
        Time ::= P2.Time

-- Imported from P7Operations (clause 10.2.5) --

        serviceNotAvailable ::= P7Operations.serviceNotAvailable
        attributeNotAvailable ::= P7Operations.attributeNotAvailable
        accessControlViolation ::= P7Operations.accessControlViolation
        mTANotAvailable ::= P7Operations.mTANotAvailable
        invalidRedispatch ::= P7Operations.invalidRedispatch
        thirdPartyError ::= P7Operations.thirdPartyError
        submitOrProbeError ::= P7Operations.submitOrProbeError
        invalidMSSequenceNumber ::= P7Operations.invalidMSSequenceNumber
        invalidRange ::= P7Operations.invalidRange
        invalidFilter ::= P7Operations.invalidFilter
        invalidLimit ::= P7Operations.invalidLimit
        noEntries ::= P7Operations.noEntries
        fetchRestrictionViolation ::= P7Operations.fetchRestrictionViolation
        MSSequenceNumber ::= P7Operations.MSSequenceNumber
        AutoCorrelationRequest ::= P7Operations.AutoCorrelationRequest
        FetchArgument ::= P7Operations.FetchArgument
        DispatchEnvelope ::= P7Operations.DispatchEnvelope
        UserReportRequest ::= P7Operations.UserReportRequest
        ExplicitConversion := P7Operations.ExplicitConversion
        DispatchResult ::= P7Operations.DispatchResult
        MSEntry ::= P7Operations.MSEntry
        MSEnvelope ::= P7Operations.MSEnvelope
        UMPDUContent ::= P7Operations.UMPDUContent
        UserReportRequest ::= P7Operations.UserReportRequest


-- Operations --

        dispatchIPMessage  OPERATION
                           ARGUMENT SEQUENCE {
                               envelope DispatchIPMEnvelope,
                               content DispatchIPMContent}
                           RESULT DispatchIPMResult
```

```
                    ERRORS {
                        noReplyToUsers,
                        oRDescriptorWithoutORName,
                        noPrimaryOrCopyRecipientsPresent,
                        noBlindCopyRecipientsPresent,
                        blindCopyRecipientsPresent,
                        serviceNotAvailable,
                        accessControlViolation,
                        mTANotAvailable,
                        invalidRedispatch,
                        thirdPartyError,
                        submitOrProbeError,
                        noEITsSpecified,
                        invalidReply,
                        invalidForwarding }
                    ::= 201

issueIPMNotification    OPERATION
                        ARGUMENT SET {
                            notificationFor [0] IMPLICIT MSSequenceNumber,
                            CHOICE {
                                [1] IMPLICIT Receipt,
                                [2] IMPLICIT NonReceipt},
                                [3] IMPLICIT UserReportRequest DEFAULT
                                        {basic},
                            autoCorrelate [29] IMPLICIT
                                AutoCorrelationRequest OPTIONAL
                                        --only allowed request is for --
                                        --delivery/non-delivery notifications--
                                        --default is no auto-correlation--}
                        RESULT DispatchIPMResult
                        ERRORS {
                            serviceNotAvailable,
                            accessControlViolation,
                            mTANotAvailable,
                            invalidMSSequenceNumber,
                            notIPMessage,
                            receiptNotRequested
                            nonReceiptNotRequested,
                            noOriginator,
                            invalidReason}
                        ::= 202

iPMFetchMessage     OPERATION
                    ARGUMENT SET {
                        COMPONENTS OF FetchArgument,
                        desiredIPMParts [30] IMPLICIT PartsSelector
                            OPTIONAL
                                --default is complete content subject to the fetch
                                --restrictions-- }
                    RESULT SET {
                        IPMMessage,
                        list [0] IMPLICIT SEQUENCE OF MSSequenceNumber
                            OPTIONAL,
                        next [1] IMPLICIT MSSequenceNumber OPTIONAL
                            --if there are more messages than the limit--}
```

```
                    ERRORS {
                        serviceNotAvailable,
                        attributeNotAvailable,
                        accessControlViolation,
                        invalidMSSequenceNumber,
                        invalidRange,
                        invalidFilter,
                        invalidLimit,
                        noEntries,
                        fetchRestrictionViolation,
                        invalidPartsSelector }
                    ::= 203

-- Data types for dispatchIPMessage operation --

        DispatchIPMEnvelope ::= SET {
            COMPONENTS OF DispatchEnvelope,
            [30] IMPLICIT DispatchIPMParameters OPTIONAL}

        DispatchIPMParameters ::= SET {
            [0] IMPLICIT ReplyParameters OPTIONAL
                --only present if a reply--,
            [1] IMPLICIT UseRecipientsFromHeading OPTIONAL
                --in addition to any recipients in the dispatch envelope--}

        ReplyParameters ::= SET {
            inReplyTo [0] IMPLICIT MSSequenceNumber,
            replyRecipients [1] IMPLICIT BIT STRING {
                replyToOriginator (0), replyToUsers (1), replyToAll (2)},
            reportRequest [2] IMPLICIT ReportRequest DEFAULT {},
            replyRequest [3] IMPLICIT BOOLEAN DEFAULT FALSE}

        UseRecipientsFromHeading ::= SET  {
            whereFrom INTEGER {
                primaryAndCopy (0), blindCopy (1)},
            [0] IMPLICIT UserReportRequest DEFAULT {basic},
            [1] IMPLICIT ExplicitConversion OPTIONAL}

        DispatchIPMContent ::= CHOICE {
            iPMessage [0] TPT-DATA {DispatchIPMessage},
            redispatch [1] IMPLICIT MSSequenceNumber
                --of non-receipt notification whose returned content is to be--
                --re-dispatched, or--
                --of non-delivery notification whose returned content is to be--
                --redispatched--}

        DispatchIPMessage SEQUENCE {
            DispatchIPMHeading,
            DispatchIPMBody}
```

```
DispatchIPMHeading ::= SET {
    -- same as P2.Heading except that all is OPTIONAL--
    IPMessageId OPTIONAL,
    originator [0] IMPLICIT ORDescriptor OPTIONAL,
    authorizingUsers [1] IMPLICIT SEQUENCE OF ORDescriptor OPTIONAL,
    primaryRecipients [2] IMPLICIT SEQUENCE OF Recipient OPTIONAL,
    copyRecipients [3] IMPLICIT SEQUENCE OF Recipient OPTIONAL,
    blindCopyRecipients [4] IMPLICIT SEQUENCE OF Recipient OPTIONAL,
    inReplyTo [5] IMPLICIT IPMessageId OPTIONAL,
    obsoletes [6] IMPLICIT SEQUENCE OF IPMessageId OPTIONAL,
    crossReferences [7] IMPLICIT SEQUENCE OF IPMessageId OPTIONAL,
    subject [8] Subject OPTIONAL,
    expiryDate [9] IMPLICIT Time OPTIONAL,
    replyBy [10] IMPLICIT Time OPTIONAL,
    replyToUsers [11] IMPLICIT SEQUENCE OF ORDescriptor OPTIONAL,
    importance [12] IMPLICIT Importance DEFAULT normal,
    sensitivity [13] IMPLICIT Sensitivity OPTIONAL,
    obsoletingTimes [16] IMPLICIT SEQUENCE OF ObsoletingTime
        OPTIONAL}

DispatchIPMBody ::= SEQUENCE OF CHOICE {
    bodyPart [0] TPT-DATA {BodyPart}
    forward [1] IMPLICIT MSSequenceNumber
        --of IP-message to be forwarded, or of non-receipt notification--
        --whose returned content is to be forwarded, or of non-delivery--
        --notification whose returned (IP-message) content is to be--
        --forwarded--}

DispatchIPMResult ::= SET {
    COMPONENTS OF DispatchResult,
    IPMessageId}

-- Data types for issueIPMNotification operation --

Receipt ::= SET {}

NonReceipt ::= SET {
    reason [0] IMPLICIT NonReceiptReason,
    nonReceiptQualifier [1] IMPLICIT NonReceiptQualifier OPTIONAL}

-- Data types for iPMFetchMessage operation --

PartsSelector ::= SEQUENCE OF
    CHOICE {
        part [0] IMPLICIT PartNumber,
        range [1] IMPLICIT SEQUENCE {
            from [0] IMPLICIT PartNumber DEFAULT 0,
            to [1] IMPLICIT PartNumber OPTIONAL
                --default is no upper bound--},
        explodeForwardedPart [2] IMPLICIT PartsSelector
            --only present when exploding a forwarded body part--}

PartNumber ::= INTEGER
    --the body parts at any level of nesting are implicitly numbered--
    --sequentially from 1;--
    --the heading is considered to be part number 0--
```

```
IPMMessage ::= SEQUENCE {
    entry [0] IMPLICIT MSEntry OPTIONAL,
        --absent if "entry" not requested;--
        --MS Status, if returned, is that prior to the fetch-,
    envelope [1] MSEnvelope OPTIONAL
        --absent if "envelope" not requested--,
    content [2] IPMContent OPTIONAL
        --absent if neither delivered content nor returned content--
        --requested--
        --or (for a non-delivery report) if no content has been returned--}

IPMContent ::= CHOICE {
    complete TPT-DATA {UMPDUContent}--of content type P2--,
    selectedParts [30] IMPLICIT SEQUENCE OF Part}

Part ::= SEQUENCE {
    part [0] IMPLICIT PartNumber,
    CHOICE {
        heading [1] TPT-DATA {Heading},
        bodypart [2] TPT-DATA {BodyPart},
        explodedForwardedPart [3] IMPLICIT SEQUENCE OF Part}}

-- Errors --

noReplyToUsers ERROR
    PARAMETER SET {}
    ::= 201

oRDescriptorWithoutORName ERROR
    PARAMETER SET {SEQUENCE OF ORDescriptor
        --not necessarily a complete list--}
    ::= 202

noPrimaryOrCopyRecipientsPresent ERROR
    PARAMETER SET {}
    ::= 203

noBlindCopyRecipientsPresent ERROR
    PARAMETER SET {}
    ::= 204

blindCopyRecipientsPresent ERROR
    PARAMETER SET {}
    ::= 205

receiptNotRequested ERROR
    PARAMETER SET {}
    ::= 206

nonReceiptNotRequested ERROR
    PARAMETER SET {}
    ::= 207

noOriginator ERROR
    PARAMETER SET {}
    ::= 208

invalidReason ERROR
    PARAMETER SET {}
    ::= 209
```

```
noEITsSpecified ERROR
    PARAMETER SET {}
    ::= 210

invalidPartsSelector ERROR
    PARAMETER SET {}
    ::= 211

invalidForwarding ERROR
    PARAMETER SET {SEQUENCE OF IPMessageProblem
        --not necessarily a complete list--}
    ::= 212

IPMessageProblem ::= SEQUENCE {
    reason [0] IMPLICIT INTEGER {
        invalidMSSequenceNumber (0), notIPMessage (1) },
    [1] IMPLICIT MSSequenceNumber --of message at fault--}

invalidReply ERROR
    PARAMETER SET {SEQUENCE OF IPMessageProblem
        --not necessarily a complete list--}
    ::= 213

END --of module P7ipmOperations --
```

10.3.5 Object Identifiers for IPM

10.3.5.1 Attribute Types for IPM

The Attributes defined in clause 10.3.6 correspond to the IPM-specific Mailbox Attributes of Tables 3 and 4.

The object identifier component ipmAttribute is defined by:

```
ipmAttribute OBJECT IDENTIFIER ::= {ecma-xxx 101}
```

10.3.5.2 Action Types for IPM

The Actions defined in clause 10.3.6 correspond to the IPM-specific Actions of Table 8.

The object identifier component ipmAction is defined by:

```
ipmAction OBJECT IDENTIFIER ::= {ecma-xxx 102}
```

10.3.5.3 Return Classes for IPM

The Return Classes defined in clause 10.3.6 correspond to the IPM Auto-correlation Request services described in clause 9.3.3

The object identifier component ipmReturnClass is defined by:

```
ipmReturnClass OBJECT IDENTIFIER ::= {ecma-xxx 103}
```

10.3.6 The P7ipmIdentifiers Module

This clause is a formal definition of the p7ipmIdentifiers module.

```
P7ipmIdentifiers DEFINITIONS ::=
BEGIN
```

```
-- Imported from ECMA-93 --

        EncryptedParameters ::= P2.EncryptedParameters
        G3FaxParameters ::= P2.G3FaxParameters
        IA5Parameters ::= P2.IA5Parameters
        ISO6937Parameters ::= P2.ISO6937Parameters
        NonReceiptInformation ::= P2.NonReceiptInformation
        ReceiptInformation ::= P2.ReceiptInformation
        TTXParameters ::= P2.TTXParameters
        VideotexParameters ::= P2.VideotexParameters
        VoiceParameters ::= P2.VoiceParameters

-- Imported from P7Operations (clause 10.2.5) --

        EITs ::= P7Operations.EITs
        MSStatus ::= P7Operations.MSStatus

-- Imported from 7ipmOperations (clause 10.3.4) --

        DispatchSequenceNumber ::= P7ipmOperations.DispatchSequenceNumber
        Importance ::= P7ipmOperations.Importance
        IPMessageId ::= P7ipmOperations.IPMessageId
        ObsoletingTime ::= P7ipmOperations.ObsoletingTime
        ORDescriptor ::= P7ipmOperations.ORDescriptor
        Sensitivity ::= P7ipmOperations.Sensitivity
        Subject ::= P7ipmOperations.Subject
        Time ::= P7ipmOperations.Time

-- Imported from P7Identifiers (clause 10.2.7) --

        ATTRIBUTE ::= P7Identifiers.ATTRIBUTE

        ACTION ::= P7Identifiers.ACTION
        RETURN-CLASS ::= P7Identifiers.RETURN-CLASS

        ecma-xxx ::= P7Identifiers.ecma-xxx

-- Attributes --

        ipmAttribute OBJECT IDENTIFIER ::= {ecma-xxx 101}

        p2MessageType ATTRIBUTE
            REPRESENTED BY IPMMessageType
            ::= {ipmAttribute 1}

        IPMMessageType ::= INTEGER {
            ipMessage (0),
            receiptNotification (1),
            nonReceiptNotification (2) }

        p2IPMessageId ATTRIBUTE
            REPRESENTED BY P7ipmOperations.IPMessageId
            ::= {ipmAttribute 2}

        p2Originator ATTRIBUTE
            REPRESENTED BY P7ipmOperations.ORDescriptor
            ::= {ipmAttribute 3}

        p2Subject ATTRIBUTE
            REPRESENTED BY Subject
            ::= {ipmAttribute 4}

        p2Importance ATTRIBUTE
            REPRESENTED BY Importance
            ::= {ipmAttribute 5}
```

```
p2Sensitivity ATTRIBUTE
    REPRESENTED BY Sensitivity
    ::= {ipmAttribute 6}

p2ExpiryDate ATTRIBUTE
    REPRESENTED BY Time
    ::= {ipmAttribute 7}

p2InReplyTo  ATTRIBUTE
    REPRESENTED BY IPMessageId
    ::= {ipmAttribute 8}

p2ReplyBy ATTRIBUTE
    REPRESENTED BY Time
    ::= {ipmAttribute 9}

p2Autoforwarded ATTRIBUTE
    REPRESENTED BY BOOLEAN
    ::= {ipmAttribute 10}

p2PrimaryRecipient ATTRIBUTE
    MULTIVALUED
    REPRESENTED BY ORDescriptor
    ::= {ipmAttribute 11}

p2CopyRecipient ATTRIBUTE
    MULTIVALUED
    REPRESENTED BY ORDescriptor
    ::= {ipmAttribute 12}

p2BlindCopyRecipient ATTRIBUTE
    MULTIVALUED
    REPRESENTED BY ORDescriptor
    ::= {ipmAttribute 13}

p2AuthorizingUser ATTRIBUTE
    MULTIVALUED
    REPRESENTED BY ORDescriptor
    ::= {ipmAttribute 14}

p2Obsoletes ATTRIBUTE
    MULTIVALUED
    REPRESENTED BY IPMessageId
    ::= {ipmAttribute 15}

p2CrossReference ATTRIBUTE
    MULTIVALUED
    REPRESENTED BY IPMessageId
    ::= {ipmAttribute 16}

p2IPMSynopsis ATTRIBUTE
    REPRESENTED BY IPMessageSynopsis
    ::= {ipmAttribute 17}

IPMessageSynopsis ::= SEQUENCE OF PartSynopsis
```

```
PartSynopsis ::= SEQUENCE {
    part [0] IMPLICIT PartNumber,
    fetched [1] IMPLICIT BOOLEAN DEFAULT FALSE,
    length [2] IMPLICIT INTEGER--in octets,approximate--,
    detail [3] CHOICE {
        iA5Text [0] IMPLICIT IA5TextParameters,
        tLX [1] IMPLICIT SET {},
        voice [2] IMPLICIT VoiceParameters,
        g3Fax [3] IMPLICIT G3FaxParameters,
        tIFO [4] IMPLICIT SET {},
        tTX [5] IMPLICIT TTXParameters,
        videotex[6] IMPLICIT VideotexParameters,
        nationallyDefined [7] SET {},
        encrypted [8] IMPLICIT EncryptedParameters,
        forwardIPMessage [9] IPMessageSynopsis,
        sFD [10] IMPLICIT SET {},
        tIF1 [11] IMPLICIT SET {},
        oDA [12] IMPLICT SET {},
        iSO6937Text [13] IMPLICIT ISO6937TextParameters,
        unidentified[14] IMPLICIT SET {}}

p2ReceiptNonReceiptNotification ATTRIBUTE
    REPRESENTED BY ReceiptNonReceiptNotification
    ::= {ipmAttribute 18}

ReceiptNonReceiptNotification ::= SET {
    reported IPMessageId,
    actualRecipient [1] IMPLICIT ORDescriptor OPTIONAL,
    intendedRecipient [2] IMPLICIT ORDescriptor OPTIONAL,
    converted EITs OPTIONAL}

p2ReceiptInformation ATTRIBUTE
    REPRESENTED BY ReceiptInformation
    ::= {ipmAttribute 19}

p2NonReceiptInformation ATTRIBUTE
    REPRESENTED BY NonReceiptInformation
        --excluding the returned IM-UAPDU--
    ::= {ipmAttribute 20}

p2ReplyToUser ATTRIBUTE
    REPRESENTED BY ORDescriptor
    ::= {ipmAttribute 21}

p2ObsoletingTime ATTRIBUTE
    REPRESENTED BY ObsoletingTime
    ::= {ipmAttribute 22}

p2ContentReturned ATTRIBUTE
    REPRESENTED BY BOOLEAN
    ::= {ipmAttribute 23}

-- Action types --

ipmAction OBJECT IDENTIFIER ::= {ecma-xxx 102}

atNonReceipt ACTION
    ACTION PARAMETER IS DispatchSequenceNumber
    ::= {ipmAction 1}
```

```
        atReceipt ACTION
            ACTION PARAMETER IS DispatchSequenceNumber
            ::= {ipmAction 2}

        atReply ACTION
            ACTION PARAMETER IS DispatchSequenceNumber
            ::= {ipmAction 3}

        atForward ACTION
            ACTION PARAMETER IS DispatchSequenceNumber
            ::= {ipmAction 4}

        atIPMFetch ACTION
            ACTION PARAMETER IS MSStatus --as after the Action
            ::= {ipmAction 5}

        atAutoNonReceipt ACTION
            AUTOMATIC WITH REGISTRATION PARAMETER NULL
            ACTION PARAMETER IS DispatchSequenceNumber
            ::= {ipmAction 6}

        atAutoReceipt ACTION
            AUTOMATIC WITH REGISTRATION PARAMETER NULL
            ACTION PARAMETER IS DispatchSequenceNumber
            ::= {ipmAction 7}

        atIPMRedispatch ACTION
            ACTION PARAMETER IS DispatchSequenceNumber
            ::= {ipmAction 8}

        atAutoForward ACTION
            AUTOMATIC WITH REGISTRATION PARAMETER ForwardingParameter
            ACTION PARAMETER IS DispatchSequenceNumber
            ::= {ipmAction 9}

        ForwardingParameter ::= SET {
            recipient [0] IMPLICIT ORDescriptor,
            nrnComment [1] IMPLICIT PrintableString OPTIONAL}

        atAutoExpire ACTION
            AUTOMATIC WITH REGISTRATION PARAMETER NULL
            ACTION PARAMETER IS NULL
            ::= {ipmAction 10}

-- Return Classes --

        ipmReturnClass OBJECT IDENTIFIER ::= {ecma-xxx 103}

        rcReceiptNotification RETURN-CLASS
            AUTO-CORRELATION ON ORDescriptor
                --of primary, copy and blind copy recipients--
            CONDITIONAL -- upon Report Request --
            ::= {ipmReturnClass 1}

        rcReply RETURN-CLASS
            AUTO-CORRELATION ON ORDescriptor
                --of primary, copy and blind copy recipients--
            CONDITIONAL -- upon Reply Request --
            ::= {ipmReturnClass 2}

END --of module P7ipmIdentifiers --
```

11. USE OF LOWER LAYER SERVICES

This clause describes how the ROS (ECMA TR/31) is used by the Mailbox Client and the Mailbox Server, and how the MTS (see ECMA-93 MIDA) is used by the Mailbox server.

11.1 Mapping onto the Remote Operation Service (ROS)

11.1.1 Conformance to ECMA TR/31

The Mailbox Access Protocol defined in this Standard:

- uses the concept and notation of remote operations as defined in ECMA TR/31;
- allows any choice of mapping, defined in clause 6 of ECMA TR/31.

11.1.2 Services Provided by ROS to the Mailbox Application

ROS provides an execution environment to the Mailbox Application. To establish the object binding between a User and a single Mailbox, the execution environment provides the Mailbox Client and the Mailbox server with a ROS-association.

11.1.3 Mapping onto Service Primitives

The mapping of the operations onto the ROS service primitives is described in clause 5.5 of the ECMA TR/31.

The following additions have to be made for this mapping.

- the Application-protocol parameter of the RO-BEGIN service is given the value 7,
- the OP-class parameter in the RO-INVOKE service is given the value "Synchronous Mode with Error or Result",
- the Priority parameter in the RO-REJECT-U service is given the value 1. Note that the value of the Priority parameter in other services is not relevant because of the Synchronous mode of operation, these values may be selected locally.

11.2 Mapping onto the Message Transfer Service

11.2.1 Introduction

The Mailbox Application is closely linked to and uses the Message Transfer Service in ECMA-93 second edition.

11.2.2 Use of the Message Submission Service

When the MBC invokes one of the operations dispatchMessage, dispatchIPMessage or issueIPMNotification, this results in the MBS using MT-SUBMIT to pass the message to the MTA. Only after successful submission is the result of the operation sent back to the MBC.

The parameters of the MT-SUBMIT.request are filled with information from correspondingly named protocol elements of the operations. The DispatchSequenceNumber, which is generated by the MBS, is passed in character form in the parameter UA-content-id.

11.2.3 Use of the Probe Service

When the MBC invokes the operation dispatchProbe this results in the MBS using MT-PROBE to pass the probe request to the MTA. Only after successful acceptance of the probe request is the result of the operation sent back to the MBC.

The parameters of the MT-PROBE.request are filled with information from correspondingly named protocol elements of the operation. The DispatchSequenceNumber, which is generated by the MBS, is passed in character form in the parameter UA-content-id.

11.2.4 Use of the Cancel Deferred Delivery Service

When the MBC invokes the operation cancel this results in the MBS using MT-CANCEL to pass the cancellation request to the MTA.

Only after successful acceptance of the cancellation request is the result of the operation sent back to the MBC.

11.2.5 Use of the Registration Service

When the MBC invokes the operation registerwithMTA this results in the MBS using MT-REGISTER to pass the new registration information to the MTA.

Only after successful acceptance of the new register information is the result of the operation sent back to the MBC.

11.2.6 Use of the Message Delivery Service

When the MTA issues an MT-DELIVER.indication to the MBS, the MBS takes over the message and generates an MS Message in the Message Store. In the process, all mandatory and subscribed optional attributes are generated for both the MS Entry and the Inlog Entry.

An MS Sequence Number is allocated and the MS Status value is set to "New". If any relevant automatic actions have been requested by the MBC, the MBS performs these and changes the MS Status value accordingly. Delivery of a message does not result in any immediate communication between the MBS and the MBC.

11.2.7 Use of the Message Notification Service

When the MTA issues an MT-NOTIFY.indication to the MBS, the MBS takes over the notification and generates an MS Message in the Message Store. In the process, all mandatory and subscribed optional attributes are generated for both the MS Entry and the Inlog Entry.

An MS Sequence Number is allocated and the MS Status value is set to "New". If auto-correlation has been requested for the corresponding UserMessage by the MBC as part of the Dispatch, the MBS performs the auto-correlation and changes MS Status to "Processed". Notification of a message does not result in any immediate communication between the MBS and the MBC.

APPENDIX A

DEFINITIONS FOR ASN.1 MACROS

This Appendix is an integral part of the Standard

A.1 Macro for Defining Attribute Types

The ATTRIBUTE macro is used to define an Attribute that may be transferred in the Mailbox Access Protocol. The macro is defined in ECMA TR/42.

A.2 Macro for Defining Actions

The ACTION macro is used to define an Action performed by the MBS which is recorded in the Action History Attribute associated with a message in the Message Store. It is defined as follows:

```
ACTION MACRO ::=
BEGIN
    TYPE NOTATION ::= Registration Action | Action
    VALUE NOTATION ::= value (VALUE OBJECT IDENTIFIER)
    Registration ::= "AUTOMATIC WITH REGISTRATION PARAMETER" type
    Action ::= "ACTION PARAMETER IS" type
END
```

The value notation yields the object identifier for the Action being defined; this object identifier is used for the ActionType data type (see clause 10.2.7).

The type notation defines via the Action clause the data type of the associated action parameter recorded in the Action History Attribute. The Registration clause is present if the automatic execution of the Action can be registered with the MBS; this clause also defines the data type of the associated registration parameter used to qualify the automatic action.

A.3 Macro for Defining Return Classes

The RETURN-CLASS macro is used to define an Auto-correlation Class. It is defined as follows:

```
RETURN-CLASS MACRO ::=
BEGIN
    TYPE NOTATION ::= "AUTO-CORRELATION ON" AutoCorrelationElement
        Qualifier
    VALUE NOTATION ::= value (VALUE OBJECT IDENTIFIER)
    AutoCorrelationElement ::= type
    Qualifier ::= "CONDITIONAL" | empty
END
```

The value notation yields the object identifier for the return class being defined; this object identifier is used for the ReturnClass data type (see clause 10.2.7).

The type notation defines the data type of the Auto-correlation Element (which must be multivalued) against which the auto-correlation is performed. If the auto-correlation is not performed unconditionally for every value of the Auto-correlation Element, "CONDITIONAL" is present; the conditions under which the auto-correlation is performed are stated in an associated comment.

A.4 Macro for Defining Third Party Transfer Objects

The TPT-DATA macro is used to define data types for "objects" which may be represented by either the object itself ("by value") or by a Third Party Transfer reference ("by reference"). The macro is defined in ECMA TR/42. As the Third Party Transfer Application is not yet defined in detail, this version of the Standard does not include support of the "by reference" case.

APPENDIX B

SUMMARY OF APPLICATION-WIDE IDENTIFIERS

This Appendix is not part of the Standard

Module ID	Value	Name
P1	0	ORName
P1	1	CountryName
P1	2	AdministrationName
P1	3	GlobalDomainIdentifier
P1	4	MPDUIdentifier
P1	5	EncodedInformationTypes
P1	6	ContentType
P1	7	Priority
P1	8	PerMessageFlag
P1	9	TraceInformation
P1	10	UAContentID
P2	11	IPMessageId
P7Operations	28	AttributeValue
P7Operations	29	ObjRequest
P1	30	InternalTraceInfo

APPENDIX C

USE OF OTHER APPLICATIONS FOR SUPPORT OF THE MAILBOX APPLICATION

This Appendix is not part of the Standard

The Mailbox Application is one of the Distributed Office Applications as described in the ECMA/TR on Framework for Distributed Office Applications. This Appendix describes the other Distributed Office Applications which are essential to allow satisfactory operation of a Mailbox Server as described in clauses 2 and 3 of this Standard. In general, the other application functionality can potentially be provided by means other than the explicit Distributed Office Applications mentioned below.

Most of the Mailbox services are provided over the Mailbox Access Protocol as described in clause 10 of this Standard. However, some Mailbox services as well as some additional services of a more administrative character are provided by various other Distributed Office Applications and their respective protocols. This clause describes which services are provided by which application and also gives the additional definitions needed within that application in order to provide these services.

The Supportive Applications which are used by the Mailbox Server are:

- Directory Application,
- Third Party Transfer.

The use of these applications is described in more detail in the following clauses.

C.1 Use of the Directory Application

The Directory is used to store and maintain various data associated with the administrative functions needed by the Mailbox Server.

C.1.1 Directory Object Classes Needed by the Mailbox Server

The Object Classes needed in the Directory for administration of the Mailbox Server fall into three categories:

- Mailbox Users: A number of different Directory Object Classes can potentially be Owners or Subsidiary Users of a Mailbox. (For more details see clause C.1.1.1 below).
- Mailboxes: This is an object class which is specific to the Mailbox Server. Most of the administrative information is stored in this class of object. (For more details see clause C.1.1.2 below).
- Mailbox Server: This type of object belongs to the object class Application Entity. (For more details see clause C.1.1.3 below).

Note C.1

The formal definitions of Object Classes and Directory Attributes mentioned in the following clauses are still to be defined.

C.1.1.1 Mailbox Users

Many different object classes can be allowed as Owners or Subsidiary Users of Mailboxes. The most common case is that a human being (object classes Organizational Person, Organizational Role or Residential Person) appears as User, but groups (object classes Organization, Organization Unit or Group of Names) or computer applications (object class Application Entity) can also be valid Users. This is not a complete list, it will need to be flexible for future extensions. The only restriction is that an object of class Mailbox cannot be a Mailbox User. This restriction has been introduced in order to prevent looping of ownerships and access rights.

An object entry for a Mailbox Owner should contain the following Directory Attribute:

- Mailbox name : This is used to link the Mailbox Owner entry and the Mailbox entry. An Owner can have more than one Mailbox, which means that several Mailbox names are allowed.

C.1.1.2 Mailboxes

The Mailbox object class is specific to the Mailbox Server. The important features of this object class are:

- It has an O/R Name as the Distinguished Name of the object. (This is one of the defined variants of Form 1 in ECMA-93. Which variant is decided on a per Naming Domain basis by the Naming Authority).
- It contains a list of the Mailbox Owner and the Subsidiary Users with their access rights.
- It contains the name of the Mailbox Server in which the Mailbox is located.
- It contains the list of the subscribed optional Mailbox Services.
- It contains the list of the subscribed optional Mailbox Attributes.
- It contains the Status information for the Automatic Mailbox Services.
- It contains the UA Profile parameters.
- It contains information about requested Redirection of Incoming messages.

The information above is stored as Directory Attributes.

C.1.1.3 Mailbox Server

The Mailbox Server belongs to the Application Entity object class. The important features of this object are :

- It has a Distinguished Name according to the rules layed down by ISO/CCITT.
- It contains an Attribute which identifies it as a Mailbox Server.
- It contains a list of the Mailboxes (O/R Names) which are administered by the Mailbox Server.
- It contains the PSAP Address of the Mailbox Server.

The information above is stored as Directory Attributes.

C.2 Use of the Third Party Transfer

The Mailbox Server may use a colocated Third Party Transfer Agent in order to produce objects for later transfer to entity other than the MBC, if requested in a Mailbox Access Protocol operation to do so. A number of operations in the Mailbox Access Protocol contain the possibility to invoke the use of the Third Party Transfer facility e.g. the Fetch Message operation. The Mailbox Server may also use Third Party Transfer to consume an object (that is having it transferred) which has been previously produced by another entity e.g. in a dispatch operation.

Third Party Transfer is not yet defined in detail. However, the Mailbox service description and the Mailbox Access protocol specification has made provisions so as to be able to use Third Party Transfer as soon as it is fully specified.

C.2.1 Produce Third Party Object

This service allows the Mailbox Server to initiate the production of an object. This object can later be consumed by another Distributed Office Application via Third Party Transfer.

C.2.2 Consume Third Party Object

This service allows the Mailbox Server to consume an object using Third Party Transfer. The consumed object must have been produced by another Distributed Office Application Server.