

INTERNATIONAL STANDARD

ISO/IEC 10021-5

First edition
1990-12-01

Information technology — Text Communication — Message-Oriented Text Interchange Systems (MOTIS) —

Part 5: Message Store: Abstract Service Definition

*Technologies de l'information — Communication de texte — Systèmes d'échange
de texte en mode message —*

Partie 5: Dépôt de message: Définition de service abstrait



Reference number
ISO/IEC 10021-5:1990 (E)
AVA Documents
Document Sharing Hub

Contents

	Page
Foreword	iv
Introduction	v
Section one - General	
1 Scope	1
2 Normative references	2
3 Definitions	4
4 Abbreviations	9
5 Conventions	10
Section two - Message Store abstract-service definition	
6 Message Store model	13
7 Abstract-bind and Abstract-unbind-operations	23
8 Abstract-operations	27
9 Abstract-errors	42
Section three - General-attribute-types and general-auto-action-types	
10 Overview	47
11 General-attribute-types	47
12 General-auto-action-types	64
Section four - Procedures for Message Store and ports realization	
13 Overview	68
14 Consumption of the Message Transfer System abstract-service	68
15 Supply of the Message Store abstract-service	74
16 Ports realization	81

© ISO/IEC 1990
All rights reserved. No part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from the publisher.
International Organization for Standardization
Case postale 56 • CH-1211 Genève 20 • Switzerland
Printed in Switzerland

ANNEXES

A Formal assignment of object identifiers 82

B Formal definition of the Message Store abstract-service 84

C Formal definition of general-attribute-types 93

D Formal definition of general-auto-action-types 99

E Formal definition of MS parameter upper bounds 101

F Example of the Summarize abstract-operation 102

**G Differences between the CCITT Recommendation X.413 text and
 ISO/IEC 10021-5 text 104**

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

International Standard ISO/IEC 10021-5 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*.

ISO/IEC 10021-5 consists of the following parts, under the general title: *Information technology — Text Communication — Message-Oriented Text Interchange Systems (MOTIS)* —

- *Part 1: System and Service Overview*
- *Part 2: Overall Architecture*
- *Part 3: Abstract Service Definition Conventions*
- *Part 4: Message Transfer System: Abstract Service Definition and Procedures*
- *Part 5: Message Store: Abstract Service Definition*
- *Part 6: Protocol Specifications*
- *Part 7: Interpersonal Messaging System*

Annexes A, B, C and D form an integral part of this part of ISO/IEC 10021. Annexes E, F and G are for information only.

Introduction

This part of ISO/IEC 10021 is one of a number of parts of ISO/IEC 10021 (the International Standard for Message Oriented Text Interchange System (MOTIS)).

MOTIS provides for the exchange of messages between users on a store-and-forward basis. A message submitted by one user (the originator) is transferred through the message-transfer-system (MTS) and delivered to one or more other users (the recipients).

This part of ISO/IEC 10021 defines the Message Store abstract-service (MS abstract-service) which supports message-retrieval from a Message Store (MS) and indirect-message-submission through the MS in a Message Handling System (MHS). The MS abstract-service also provides message-administration services, as defined by the Message Transfer System (MTS) abstract-service.

This International Standard has been produced by joint CCITT-ISO/IEC agreement. The corresponding CCITT Recommendation is X.413 (1988). Annex G list the differences between the two documents.

Information technology — Text Communication — Message-Oriented Text Interchange Systems (MOTIS) —

Part 5: Message Store: Abstract Service Definition

Section one - General

1 Scope

This part of ISO/IEC 10021 defines the Message Store abstract-service. This abstract-service is provided by the Message Store access-protocol (specified in ISO/IEC 10021-6) in conjunction with the MTS abstract-service (defined in ISO/IEC 10021-4), together with the Remote Operations Service Element (ROSE) services (defined in ISO/IEC 9072-1). The abstract-syntax-notation for the application-layer protocols used in this part of ISO/IEC 10021 is defined in ISO 8824.

Other parts of ISO/IEC 10021 define other aspects of the MHS. ISO/IEC 10021-1 defines the user-oriented services provided by the MHS. ISO/IEC 10021-2 provides an architectural overview of the MHS. ISO/IEC 10021-3 provides a description of the abstract-service definition conventions used in MHS. ISO/IEC 10021-7 defines the abstract-service for interpersonal messaging and defines the format of interpersonal-messages.

Section two of this part of ISO/IEC 10021 contains the Message Store abstract-service definition. Clause 6 describes the MS model. Clause 7 specifies the abstract-syntax-notation for the abstract-bind and the abstract-unbind-operations. Clause 8 specifies the abstract-syntax-notation for the operations of the abstract-service. Clause 9 specifies the abstract-syntax-notation for the errors of the abstract-service.

Section three of this part of ISO/IEC 10021 defines the general-attribute-types and general-auto-action-types related to the MS. Clause 10 contains an overview. Clause 11 specifies the abstract-syntax-notation for the general-attribute-types. Clause 12 specifies the abstract-syntax-notation for the general-auto-action-types.

Section four of this part of ISO/IEC 10021 describes the procedures for Message Store and the ports realization. Clause 13 contains an overview. Clause 14 describes how the Message Store abstract-service is supplied. Clause 15 describes how the Message Transfer System abstract-service is consumed. Clause 16 describes how the MS ports are realized.

No requirement is made for conformance to this part of ISO/IEC 10021.

2 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of the part of ISO/IEC 10021. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this part of ISO/IEC 10021 are encouraged to investigate the possibility of applying the most recent editions of the standards listed below. Members of IEC and ISO maintain registers of currently valid International Standards.

2.1 Reference Model references

This part of ISO/IEC 10021 cites the following reference model related documents

ISO 7498:1984, *Information processing systems - Open Systems Interconnection - Basic Reference Model.*

2.2 Presentation references

This part of ISO/IEC 10021 cites the following presentation related documents

ISO 8824:1987, *Information processing systems - Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1).*

ISO 8824:1987/Add.1: ¹⁾ *Information processing systems - Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1) - Addendum 1: ASN.1 Extensions.*

2.3 Remote Operations references

This part of ISO/IEC 10021 cites the following remote operations related document

ISO/IEC 9072-1:1989, *Information processing systems - Text Communication - Remote Operations Part 1: Model, Notation and Service Definition.*

2.4 Directory References

This part of ISO/IEC 10021 cites the following directory related documents

ISO/IEC 9594-1: ¹⁾ *Information processing systems - The Directory - Overview of Concepts, Models and Services.*

ISO/IEC 9594-2: ¹⁾ *Information processing systems - The Directory - Models.*

ISO/IEC 9594-3: ¹⁾ *Information processing systems - The Directory - Abstract Service Definition.*

ISO/IEC 9594-4: ¹⁾ *Information processing systems - The Directory - Procedures for Distributed Operation.*

ISO/IEC 9594-5: ¹⁾ *Information processing systems - The Directory - Protocols Specifications.*

ISO/IEC 9594-6: ¹⁾ *Information processing systems - The Directory - Selected Attribute Types.*

¹⁾ To be published

- ISO/IEC 9594-7: 1) *Information processing systems - The Directory - Selected Object Classes.*
- ISO/IEC 9594-8: 1) *Information processing systems - The Directory - Authentication Framework.*

2.5 Message Handling references

This International Standard cites the following ISO/IEC message handling documents

- ISO/IEC 10021-1:1989 *Information processing systems - Text Communication - Message Oriented Text Interchange System (MOTIS) - Part 1: System and Service Overview.*
- ISO/IEC 10021-2:1989 *Information processing systems - Text Communication - Message Oriented Text Interchange System (MOTIS) - Part 2: Overall Architecture.*
- ISO/IEC 10021-3:1989 *Information processing systems - Text Communication - Message Oriented Text Interchange System (MOTIS) - Part 3: Abstract Service Definition Conventions.*
- ISO/IEC 10021-4:1989 *Information processing systems - Text Communication - Message Oriented Text Interchange System (MOTIS) - Part 4: Message Transfer System - Abstract Service Definition and Procedures.*
- ISO/IEC 10021-6:1989 *Information processing systems - Text Communication - Message Oriented Text Interchange System (MOTIS) - Part 6: Protocol Specifications.*
- ISO/IEC 10021-7:1989 *Information processing systems - Text Communication - Message Oriented Text Interchange System (MOTIS) - Part 7: Interpersonal Messaging System.*

1) To be published

3 Definitions

3.1 Common Definitions for MHS

For a list of the common definitions for MHS refer to ISO/IEC 10021-2.

3.2 Message Store Definitions

For the purpose of this part of ISO/IEC 10021 the following definitions apply

- 3.2.1 abstract-association:** An abstract binding between two communication partners, in this part of ISO/IEC 10021 the binding between a UA and an MS for the provision of the MS abstract-service, or between an MS and an MTA for the provision of the MTS abstract-service.
- 3.2.2 abstract-bind-parameters:** Parameters defined in this part of ISO/IEC 10021 which are contained in the abstract-bind operation.
- 3.2.3 abstract-unbind-parameters:** Parameters defined in this part of ISO/IEC 10021 which are contained in the abstract-unbind operation.
- 3.2.4 Administration Port:** The port offering the administration (for MTS) set of abstract-services within the MS abstract-service.
- 3.2.5 Alert abstract-operation:** An abstract-operation which allows the MS to signal, based on selection criteria, to the UA that messages or reports are waiting in the MS.
Can only be issued on an existing abstract-association.
- 3.2.6 attribute:** The information of a particular type appearing in an entry in an information-base.
- 3.2.7 attribute-type:** That component of an attribute which indicates the class of information given by that attribute.
- 3.2.8 attribute-value:** A particular instance of that class of information indicated by an attribute type.
- 3.2.9 attribute-value-assertion:** A proposition, which may be true, false, or undefined, concerning the values of attributes in an entry.
- 3.2.10 auto-action:** Actions, that can be performed automatically by the MS, based on previously registered information from the MS-owner via the UA.
- 3.2.11 auto-action-type:** An auto-action-type is used to indicate the type of auto-action, e.g. Alert.
- 3.2.12 auto-alert:** Auto-alert is the auto-action within the MS, which triggers an Alert abstract-operation or another action by the MS.
- 3.2.13 auto-forward:** Auto-forward is the auto-action within the MS, which triggers a message to be auto-forwarded to another recipient (or other recipients) by the MS.

- 3.2.14 child-entry:** An entry, other than the main-entry in an information-base. The parent-entry for a child-entry can be either the main-entry or another child-entry, depending on the number of entry levels in each case.
- 3.2.15 child-sequence-number:** A sequence-number in a parent-entry pointing to a child-entry. A parent-entry can have more than one child-sequence-number value, depending on the number of child-entries.
- 3.2.16 conditional (C) component:** An ASN.1 element which shall be present in an instance of its class as dictated by this International Standard.
See **grade**.
- 3.2.17 content-length:** An attribute which gives the length of the content of a delivered-message (or returned-content).
- 3.2.18 content-returned:** An attribute which signals that a delivered-report (or a delivered-message) contained a returned content.
- 3.2.19 converted EITs:** An attribute identifying the encoded-information-types of the message content after conversion.
- 3.2.20 creation-time:** An attribute which gives the creation-time (by the MS) of an entry.
- 3.2.21 Delete abstract-operation:** An abstract-operation used to delete one or more entries from an information-base.
- 3.2.22 delivered-EITs:** A multi-valued attribute, giving information about EITs in a delivered-message.
- 3.2.23 delivered-message entry:** An entry in the stored-messages information-base resulting from a delivered-message.
- 3.2.24 delivered-report entry:** An entry in the stored-messages information-base resulting from a delivered-report.
- 3.2.25 entry:** An information set in an information-base.
See **main-entry**, **parent-entry** and **child-entry** for further classification of entries.
- 3.2.26 entry-information:** A parameter, used in abstract-operations, which conveys selected information from an entry.
- 3.2.27 entry-information-selection:** A parameter, used in abstract-operations, which indicates what information from an entry is being requested.
- 3.2.28 entry-status:** An attribute giving information about the processing status of that entry. Possible values are **new**, **listed** or **processed**.
- 3.2.29 entry-type:** An attribute which signals if an entry is associated with a delivered-message or a delivered-report.
- 3.2.30 Fetch abstract-operation:** An abstract-operation which allows one entry to be fetched from the stored-messages information-base.

- 3.2.31 fetch-restrictions:** Restrictions, imposed by the UA, on what kind of messages it is prepared to receive as a result of fetch. The possible restrictions are on message-length, content-types and EITs.
- 3.2.32 filter:** A parameter, used in abstract-operations, to test a particular entry in an information-base and is either satisfied or not by that entry.
- 3.2.33 filter-item:** An assertion about the presence or value(s) of an attribute of a particular type in an entry under test. Each such assertion is either true or false.
- 3.2.34 forwarding-request:** This is a parameter that may be present in a Message-submission abstract-operation, invoked by the UA, to request that a message is forwarded from the MS.
- 3.2.35 general-attribute:** A set of MS attributes which are valid for all types of messages and reports, independent of content-type.
Only these MS attributes are explicitly defined in this part of ISO/IEC 10021.
- 3.2.36 general-auto-action:** Auto-actions which are valid for all types of messages and reports, independent of content-type.
Only these auto-actions are explicitly defined in this part of ISO/IEC 10021.
- 3.2.37 grade:** Defined in ISO/IEC 10021-2.
- 3.2.38 Indirect-submission Port:** The port offering the Indirect-submission abstract-service within the MS abstract-service.
The Indirect-submission abstract-service offers the same services as the Message-submission abstract-service (from the MTS abstract-service) with the added functionality of forwarding messages residing in the MS.
- 3.2.39 information-base:** Objects within the MS which store information relevant to the MS abstract-service, e.g. the stored-messages information-base, which stores the messages and reports that have been delivered into the MS.
- 3.2.40 information-base-type:** The type of information-base, e.g. the stored-messages.
- 3.2.41 limit:** A component in the selector parameter which identifies the maximum number of selected entries to be returned in the result of an abstract-operation.
- 3.2.42 list abstract-operation:** An abstract-operation which allows a selection of entries from an information-base and requested attribute information to be returned for those entries.
- 3.2.43 listed:** An entry-status value.
- 3.2.44 macro:** See ISO 8824.
- 3.2.45 main-entry:** For each successful abstract-operation which creates information-base entries, there is always one main-entry. Further, or more detailed, information resulting from the same abstract-operation can be stored in child-entries.
- 3.2.46 mandatory (M) component:** An ASN.1 element which shall always be present in an instance of its class.
See **grade**.

- 3.2.47 matching:** The process of comparing the value supplied in an attribute-value-assertion with the value of the indicated attribute-type stored in the MS or deciding whether the indicated attribute-type is present.
- 3.2.48 Message Retrieval Service Element (MRSE):** The application-service-element by means of which a receiving UA effects retrieval of messages from an MS, or any of various related tasks.
- 3.2.49 MS:** Message Store, also used as a shorter form for "MS abstract-service-provider".
- 3.2.50 MS abstract-service:** The set of capabilities that the MS offers to its users by means of its ports.
- 3.2.51 MS abstract-service-user:** The user of the MS abstract-service.
This is the UA.
- 3.2.52 MS abstract-service-provider:** The MS which provides the MS abstract-service.
- 3.2.53 MS-user:** A shorter form for "MS abstract-service-user".
- 3.2.54 Message-submission abstract-operation:** An abstract-operation which allows the UA to submit a message to the MTS via the MS, and/or to forward a message from the MS to the MTS.
- 3.2.55 multi-valued attribute:** An attribute which can have several values associated with it.
- 3.2.56 new:** An entry-status value.
- 3.2.57 optional (O) component:** An ASN.1 element which shall be present in an instance of its class at the discretion of the object (e.g.user) supplying that instance.
See **grade**.
- 3.2.58 original-EITs:** An attribute identifying the original encoded-information-types of the message content.
- 3.2.59 override:** A component of the selector parameter indicating that the previously registered-restrictions for this abstract-operation should not apply for this instance of this abstract-operation.
- 3.2.60 parent-entry:** A parent-entry has one or more child-entries, which were created as a result of the same abstract-operation.
If a parent-entry is not a child-entry of another parent-entry, it is a main-entry.
- 3.2.61 parent-sequence-number:** A sequence-number in a child-entry pointing to its parent-entry.
There can only be one parent-sequence-number in a child-entry.
- 3.2.62 partial-attribute-request:** A component of the entry-information-selection which enables the return of only selected values of a multi-valued attribute.
- 3.2.63 position:** Positions are parameters used to specify a bound of a range.
- 3.2.64 processed:** An entry-status value.
- 3.2.65 range:** A parameter, used in abstract-operations, to select a contiguous sequence of entries from an information-base.

- 3.2.66 Register-MS abstract-operation:** An abstract-operation which allows the UA to register certain information, relevant to the UA-MS interworking, in the MS .
- 3.2.67 registration:** Information which is registered in the MS and stored (until changed by the Register-MS abstract-operation) between abstract-associations.
See Register-MS abstract-operation.
- 3.2.68 registration-identifier:** An identifier for one particular set of registration-parameters for an auto-action-type.
- 3.2.69 Retrieval Port:** The port offering the retrieval set of abstract-services within the MS abstract-service.
- 3.2.70 returned-content entry:** An entry-type in the stored-messages information-base which contains the returned content from a previously submitted message.
- 3.2.71 selector:** A parameter, used in abstract-operations, to select entries from an information-base.
- 3.2.72 sequence-number:** An attribute which uniquely identifies an entry.
Sequence-numbers are allocated in ascending order.
- 3.2.73 single-valued attribute:** An attribute which can only have one value associated with it.
- 3.2.74 span:** A component in the Summarize abstract-operation result containing the lowest and highest sequence-numbers of the entries that matched the selection criteria.
- 3.2.75 stored-messages:** The most important information-base in this part of ISO/IEC 10021, used to store entries containing messages and reports delivered by the MTS to the MS.
- 3.2.76 subscription:** A long-term agreement between the MS supplier or administrator and the MS customers (MS-owners) on the availability and use of optional MS features such as optional services and attributes.
This part of ISO/IEC 10021, assumes that such a mechanism is provided, but does not prescribe or offer any standardized method for how to provide this.
- 3.2.77 substring:** A filter-item used to specify a string of characters which appear (in the same given order) in a value of an attribute.
- 3.2.78 Summarize abstract-operation:** An abstract-operation which allows a quick overview of the kind and number of entries which are currently stored in an information-base.
- 3.2.79 synopsis:** A content specific attribute that may be used to show how child-entries, containing parts of the content, are related to each other and the main-entry.
The attribute has to be specified in the International Standard, which describes the content-type, e.g. see IPM-synopsis defined in ISO/IEC 10021-7.

4 Abbreviations

ASN.1:	Abstract Syntax Notation One
C:	conditional
DL:	distribution-list
EIT:	encoded-information-type(s)
M:	mandatory
M:	multi-valued
MASE:	Message Administration Service Element
MDSE:	Message Delivery Service Element
MH:	Message Handling
MHS:	Message Handling System
MOTIS:	Message Oriented Text Interchange System
MRSE:	Message Retrieval Service Element
MS:	Message Store
MT:	Message Transfer
MTS:	Message Transfer System
N:	no
O:	optional
O/R:	originator/recipient
P:	present
ROSE:	Remote Operations Service Element
S:	single-valued
UA:	User Agent
UTC:	Universal Co-ordinated Time
Y:	yes

5 Conventions

This part of ISO/IEC 10021 uses the description conventions listed in the following four clauses.

5.1 Conventions for abstract-services

This part of ISO/IEC 10021 uses the following ASN.1-based descriptive conventions for the indicated purposes

- 1) ASN.1 itself, to specify the abstract-syntax of information-bases and their components, and common data-types.
- 2) The ASN.1 PORT macro and associated abstract-service definition conventions of ISO/IEC 10021-3, to specify the Retrieval Port.
- 3) The ASN.1 ABSTRACT-BIND, ABSTRACT-UNBIND, ABSTRACT-OPERATION, and ABSTRACT-ERROR macros and associated abstract-service definition conventions of ISO/IEC 10021-3, to specify the MS abstract-service.

Whenever this part of ISO/IEC 10021 describes a class of data structure having components, each component is categorized as one of the following **grades**

- 1) **Mandatory (M):** A mandatory component shall be present in every instance of the class.
- 2) **Optional (O):** An optional component shall be present in an instance of the class at the discretion of the object (e.g user) supplying that instance.
- 3) **Conditional (C):** A conditional component shall be present in an instance of the class as dictated by this part of ISO/IEC 10021.

5.2 Conventions for attribute-types used in table 1 of clause 11

This part of ISO/IEC 10021 uses the conventions listed below in its definition of the attribute-types for the MS abstract-service.

For the column headed "Single/Multi-valued" the following values can occur

S	single-valued
M	multi-valued.

For the column headed "Support level by the MS and access UA" the following values can occur

M	mandatory
O	optional.

For the columns headed "Presence in delivered message entry", "Presence in delivered report entry", and "Presence in returned message entry", the presence of each attribute-type is described by one of the following values

P	<u>always present</u> in the entry because
	- it is mandatory for generation by the MS; or
	- it is a mandatory or defaulted parameter in the relevant abstract-operation.

- C conditionally present in the entry. It would be present because
- it is supported by the MS and subscribed to by the user and;
 - it was present in an optional parameter in the relevant abstract-operation
- always absent, otherwise.

For the columns headed "Available for List, Alert" and "Available for Summarize", the following values can occur

N	no
Y	yes.

5.3 Conventions for attribute-types used in table 2 of clause 11

This part of ISO/IEC 10021 uses the conventions listed below in its definition of the attribute-types for the MS abstract-service. Clause 11 includes the table that lists the attribute-types.

For the column headed "Single/Multi-valued" the following values can occur

S	single-valued
M	multi-valued.

For the column headed "Source generated by" the following values can occur

MD	MessageDelivery abstract-operation
MS	Message Store
RD	ReportDelivery abstract-operation.

5.4 Font conventions for text in general

Throughout this part of ISO/IEC 10021, terms are rendered in **bold** when defined, without emphasis upon all other occasions. Terms that are proper nouns are capitalized, generic terms are not. Multi-word generic terms are hyphenated.

5.5 Font conventions for ASN.1 definitions

Throughout this part of ISO/IEC 10021, ASN.1 definitions are written in a different (**bold**) font than the rest of the document in order to highlight the difference between normal text and ASN.1 definitions. The font used for ASN.1 definitions is also one size smaller than the ordinary text. When ASN.1 protocol elements and elements values are described in accompanying text, their names are rendered in **bold**.

5.6 Rules for ASN.1 definitions

ASN.1 definitions appears both in the body of the document to aid the exposition, and again, formally, in annexes for reference. If differences are found between the ASN.1 used in the exposition and that formally defined in the corresponding annex, a specification error is indicated.

Section two - Message Store abstract-service definition

6 Message Store model

The Message Store (MS) is modeled as an atomic object, which acts as a provider of services to an MS abstract-service-user (i.e. a User Agent), and as a user of services provided by the Message Transfer System (MTS).

The MS serves an intermediary role between the UA and the MTS. Its primary function is to accept delivery of messages on behalf of a single MHS end-user, and to retain them for subsequent retrieval by the end-user's UA. The MS also provides indirect message-submission and message-administration services to the UA, in effect, via "pass-through" to the MTS. This enables the MS to provide additional functionality compared to submission directly to the MTA; such as forwarding of messages residing in the MS and logging facilities.

Like the UA, the MS acts on behalf of only a single MHS end-user; i.e. it does not provide a common or shared multi-user MS service.

The MS is described using an abstract model in order to define the services provided by the MS - the Message Store abstract-service. Figure 1 shows the MS abstract-service in relation to its user and to the Message Transfer System abstract-service. In this figure, the open boxes represent the consumption of the abstract service, and the closed boxes represent the supply of the abstract service.

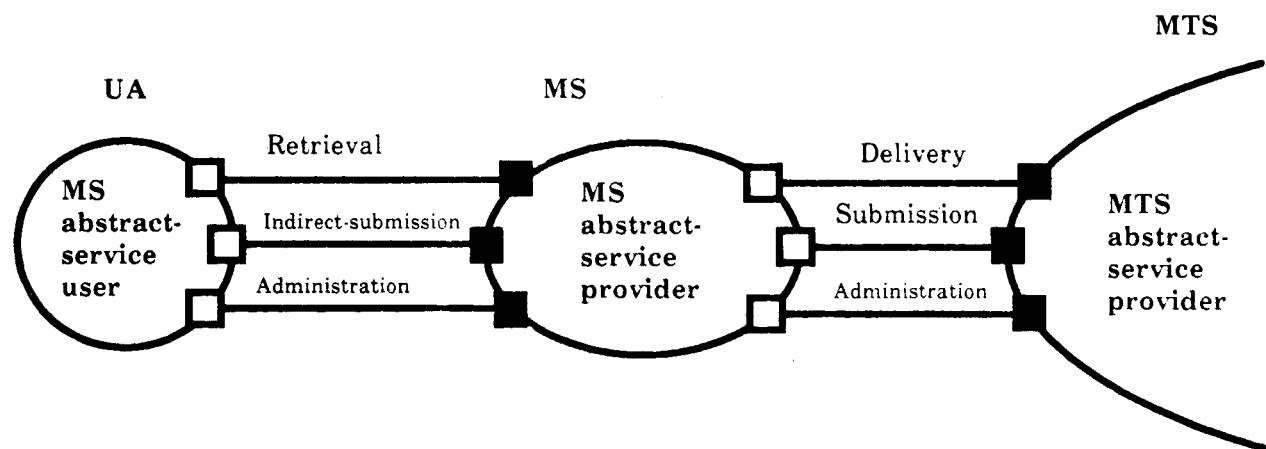


Figure 1 - Message Store abstract-service

For an introduction and description of the abstract-service concept and its definition conventions, see ISO/IEC 10021-3.

In secure messaging the MS is treated as a separate object with a unique identity and has a separate key (or a set of keys) to the UA.

6.1 Message Store object

The MS is modeled as an atomic object. It supplies the MS Retrieval Port abstract-services to the MS abstract-service-user. Acting as a "surrogate" MTS abstract-service-provider, the MS also supplies the MTS submission and administration abstract-services to the MS abstract-service-user (MS-user), and acting as a UA "surrogate", it consumes the MTS Delivery Port, Submission Port, and Administration Port abstract-services in its role as MTS abstract-service user.

The formal definition for the Message Store object is as follows:

```
mS OBJECT
  PORTS { retrieval[S],
           indirectSubmission[S],
           administration[S],
           delivery[C],
           submission[C],
           administration[C]}
  ::= id-ot-ms
```

The MS-user is also modeled as an object. It consumes the MS Retrieval Port and Indirect-submission Port abstract-services and the Administration Port abstract-services provided transparently by the MS.

```
msUser OBJECT
  PORTS { retrieval[C],
           indirectSubmission[C],
           administration[C]}
  ::= id-ot-ms-user
```

6.2 Message Store Ports

An MS provides the Retrieval, Indirect-submission, and Administration Ports to the MS abstract-service user. The collection of capabilities provided by these ports provides the MS abstract-service. The retrieval capabilities are unique to the MS. These capabilities include obtaining information on, fetching (in whole or in part), and deleting messages residing in the MS. Additional capabilities are provided for registering certain MS provided automatic actions (i.e. auto-forwarding and alert).

Additional message management services, are planned as future extensions to this part of ISO/IEC 10021. They will be performed by the MS on the UA's behalf, for logging incoming and outgoing messages, and for auto-correlating incoming notifications with logging information about outgoing messages. These extensions would form an Addendum to ISO/IEC 10021.

In order to provide the services described in 6.1 to an MS-user, the MS interacts, on behalf of the MS-user, with the MTS abstract-service, and acts as a consumer of the MTS Delivery, Submission and Administration Ports. The abstract-services provided by the MTS ports are defined in clause 8 of ISO/IEC 10021-4.

By means of the Abstract-bind operation, the MS authenticates an MS-user before providing it with any of the above retrieval capabilities. Similarly, the MTS abstract-services shall authenticate the MTS abstract-service user before extending its services to the MTS abstract-service-user.

With the exception of the Retrieval Port provided Alert service and the Indirect-submission Port provided Submission-control service, all the services provided by the MS abstract-service are invoked by the MS-user and performed by the MS.

Security-labels may be assigned to the MS in line with the security-policy in force. The security-policy may also define how security-labels are to be used to enforce the security-policy. If security-labels are assigned to the MS, the handling of stored messages and reports bearing message-security labels may be affected by the security-policy in force. If security-labels are not assigned to the MS, the handling of stored-messages and reports is discretionary.

If security-contexts are established between the UA and the MS, and between the MS and the MTA, the security-label that is assigned to a message or probe is confined by the security-context in line with the security-policy in force. If security-contexts are not established the assignment of a message-security-label to a message or probe is at the discretion of the originator.

6.2.1 Retrieval Port

The **Retrieval Port** is defined as follows:

```
retrieval PORT
  CONSUMER INVOKES {
    Summarize,
    List,
    Fetch,
    Delete,
    Register-MS}
  SUPPLIER INVOKES {
    Alert}
  ::= id-pt-retrieval
```

The details of the **Retrieval Port** abstract-services are described in clauses 7 to 9.

6.2.2 Indirect-submission Port

The **Indirect-submission Port** is defined as follows:

```
indirectSubmission PORT ::= submission
```

The **Indirect-submission Port** makes use of the Submission Port abstract-services defined in 8.2 of ISO/IEC 10021-4.

6.2.3 Administration Port

The **Administration Port** is defined in 8.4 of ISO/IEC 10021-4.

The MS shall have no interaction with the change-credentials abstract-service. If the MS-user needs to have its credentials updated, then the Register-MS abstract-operation is used. See 8.6.

6.3 Information model

This subclause describes the information model used by the MS. It models **information-bases**, which consist of **entries**, which consist of **attributes**.

6.3.1 Information-bases

The MS stores and maintains **information-bases**. An **information-base** in the MS is a "data-base" containing all the **entries** representing constituent objects of a particular category or categories.

This part of ISO/IEC 10021 defines and describes the **stored-messages information-base**. This holds information derived from message-deliveries and report-deliveries to the MS across the MTS Delivery Port, and is described in 6.4. Other information-bases are the inlog and the outlog, which are planned as a future addendum to this part of ISO/IEC 10021. These extensions are outside the scope of the corresponding CCITT Recommendation.

```
InformationBase ::= INTEGER {
    stored-messages (0),
    inlog           (1),
    outlog          (2)} (0..ub-information-bases)
```

6.3.2 Entries

Each **information-base** is organized as a sequence of **entries**. An **entry** represents a single object (such as a delivered message) within the **information-base**.

Each entry is identified by means of its **sequence-number**, unique within an **information-base**, and generated by the MS as new entries are created. Within an **information-base**, the MS generates the **sequence-numbers** in ascending order without cycling, and they are never re-used.

```
SequenceNumber ::= INTEGER (0..ub-messages)
```

NOTE - For example, the MS may choose to allocate **sequence-numbers** by using the time to a sufficient granularity to ensure uniqueness.

6.3.3 Attributes

6.3.3.1 Introduction

An **entry** consists of a set of **attributes**. This is depicted in figure 2.

Each **attribute** provides a piece of information about, or derived from, the data to which the **entry** corresponds. One such piece of information is the **sequence-number** of the **entry** itself, and another is the **creation-time**.

An **attribute** consists of an **attribute-type**, which identifies the class of information given by an **attribute**, and the corresponding **attribute-value(s)**, which are particular instances of that class appearing in the **entry**.

```
Attribute ::= SEQUENCE {
    type      AttributeType,
    values    SEQUENCE SIZE (1..ub-attribute-values) OF ANY --DEFINED BY type--}
```

NOTE - Thus, for example, in a delivered-message-entry (described in 6.4) the **attribute-type** could be the message's **priority**, and a corresponding **attribute-value** could be **urgent**.

All **attributes** in an **entry** shall be of distinct **attribute-types**.

For some **attribute-types**, an **attribute** shall only contain a single **attribute-value**. Such an **attribute-type** is said to be **single-valued**. For others, an **attribute** may contain one or more **attribute-values**, all of the same ASN.1 data-type. Such an **attribute-type** is said to be **multi-valued**. Whether an **attribute-type** is **single-valued** or **multi-valued** is stated when the **attribute-type** is defined (see 6.3.3.2).

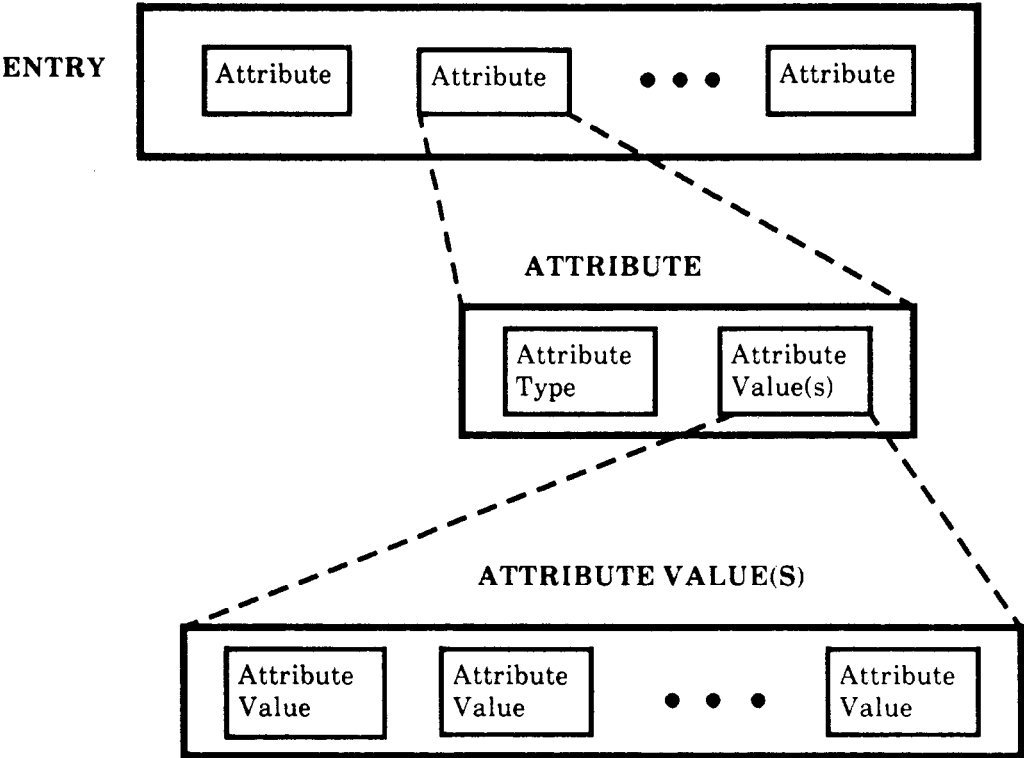


Figure 2 - The Components of an entry

NOTE - Thus, for example, the **attribute-type** for the **originator-name** attribute (described in 11.2.28) is **single-valued**, whereas that for **other-recipient-names** (described in 11.2.29) is **multi-valued**.

6.3.3.2 **Attribute-type**

Some **attribute-types** will be internationally standardized. Other **attribute-types** will be defined by national administrative authorities and private organizations. This implies that a number of separate authorities will be responsible for assigning types in a way that ensures that each is distinct from all other assigned types. This is accomplished by identifying each **attribute-type** with an object identifier when the **attribute-type** is defined .

AttributeType ::= OBJECT IDENTIFIER

Certain general-purpose **attribute-types** for the stored-messages information-base are defined in clause 11. Such **attribute-types** are known as **general-attribute-types** and attributes of these types as **general-attributes**.

6.3.3.3 **Attribute-values**

Defining an **attribute-type** also involves specifying the ASN.1 data-type to which every value in such attributes shall conform. The data-type of an **attribute-value** for the **attribute-type** is defined through the object identifier for the **attribute-type**.

6.3.3.4 **Attribute-type definition and the ATTRIBUTE Macro**

The definition of an **attribute-type** involves

- a) assigning an **object identifier** to the **attribute-type**;
- b) indicating the ASN.1 data-type of an **attribute-value**;
- c) indicating whether an attribute of this **attribute-type** may have more than one value;
- d) indicating whether an attribute of this **attribute-type** may be used for filtering using equality, substrings, and/or ordering relations, see 8.1.2.

NOTE - A filter may always test for the presence or absence in an entry of an attribute of a particular **attribute-type**.

The following ASN.1 macro is used to define an **attribute-type**. The formal definition of this macro is given in ISO/IEC 9594-2 and is documented here as an aid to the reader.

```

ATTRIBUTE MACRO      ::=
BEGIN
  TYPE NOTATION      ::=  AttributeSyntax Multivalued | empty
  VALUE NOTATION     ::=  value (VALUE OBJECT IDENTIFIER)

  AttributeSyntax    ::=  "WITH ATTRIBUTE-SYNTAX" SyntaxChoice
  SyntaxChoice       ::=  value (ATTRIBUTE-SYNTAX) Constraint | type MatchTypes

  Constraint         ::=  "(" ConstraintAlternative ")" | empty
  ConstraintAlternative ::=  StringConstraint | IntegerConstraint
  StringConstraint   ::=  "SIZE" "(" SizeConstraint ")" | empty
  SizeConstraint     ::=  SingleValue | Range
  SingleValue        ::=  value (INTEGER)
  Range              ::=  value (INTEGER) ".." value (INTEGER)
  IntegerConstraint  ::=  Range

  MatchTypes         ::=  "MATCHES FOR" Matches | empty
  Matches            ::=  Match Matches | Match
  Match              ::=  "EQUALITY" | "SUBSTRINGS" | "ORDERING"

  Multivalued        ::=  "SINGLE VALUE" | "MULTI VALUE" | empty
END

```

The correspondence between the parts of the definition, as listed above, and the various pieces of the notation introduced by the **ATTRIBUTE** macro, is as follows:

- a) **MACRO value**: The **object identifier** which is used to identify an attribute.
- b) **Attribute-syntax**: Notes which syntax-choice has been made.
- c) **Syntax-choice**: Notes whether the attribute is defined externally or internally. The syntax of all the attributes defined in this part of ISO/IEC 10021 is defined internally, which means using the choice type **MatchTypes**.
- d) **Multivalued**: Denotes whether the attribute is single or multi-valued.
- e) **Match-types**: Gives the data-type of the contents of the attribute, and describes whether the **attribute** can be matched ("**MATCHES FOR**") for equality ("**EQUALITY**"), for substrings ("**SUBSTRINGS**"), and for an ordering relation ("**ORDERING**").

Matching for this part of ISO/IEC 10021 is restricted as follows:

- i) **EQUALITY** is applicable to any attribute-syntax. The presented value shall conform to the data-type of the attribute-syntax;
- ii) **SUBSTRING** is applicable to any attribute-syntax with a **string** data type. The presented value shall be a sequence ("SEQUENCE OF"), each of whose elements conforms to the data-type, and
- iii) **ORDERING** is applicable to any attribute-syntax for which a rule can be defined that will allow a presented value to be described as less than equal to, or greater than a target value. The presented value shall conform to the data-type of the attribute-syntax.
MS uses this for the **INTEGER** and **UTCTime** data types. For **UTCTime**, the ordering is chronological, not alphabetical.

The remaining choices and parameters of the **ATTRIBUTE** macro are not used in this part of ISO/IEC 10021.

6.3.4 Main-entries, parent-entries, and child-entries

Although entries in a single information-base are generally independent of each other, the MS information model allows such entries to be related to one another. One entry, a **child-entry**, may be the child of another, its **parent-entry**, in a tree-structured relationship. An entry which is not a **child-entry** is termed a **main-entry**.

This relationship is recorded by means of two special **general-attributes**:

- a) **parent-sequence-number**: This single-valued attribute gives the sequence-number of a **child-entry**'s **parent-entry**. It is absent from a **main-entry**. Its definition is given in 11.2.30.
- b) **child-sequence-numbers**: This multi-valued attribute gives the sequence-numbers of all the **child-entries** of a **parent-entry**. It is absent from an entry which is not a **parent-entry**. Its definition is given in 11.2.1.

The abstract-operations of the MS abstract-service (see clause 8) act by default only on **main-entries**. Some may be directed to act on all entries, both **main-entries** and **child-entries**. In particular, the argument of a Delete abstract-operation (see 8.5) may only select **main-entries**, in which case the **main-entry** and all its children and children's children, etc, will also be deleted.

NOTE - This concept allows, for example, that body-parts of an Interpersonal Message which contain a forwarded message (for details see 19.1 of ISO/IEC 10021-7) to be represented by individual **child-entries**. The **content general-attribute** of the **main-entry** will comprise the complete **content**, so the data representing that message **body-part** is logically present in more than one entry.

6.4 Stored-messages

The **stored-messages information-base** acts as a repository for information obtained from the MessageDelivery and ReportDelivery abstract-operations of the Delivery Port. It contains entries for delivered messages (**delivered-message-entries**), of an open-ended number of content-types, and for reports (**delivered-report-entries**). An entry in the **stored-messages information-base** is created by the MS when a message or report is delivered to the MS. For more details of these entries and how they are generated, see clauses 11 and 15.

To draw information from the content of a message, the MS must know the content's syntax and semantics, as signaled via the content-type. In general, a particular instance of the MS has knowledge of

zero or more content-types. When an MS encounters a message of whose content-type it has insufficient knowledge, it is unable to generate any content-type-specific attributes in the message's entry.

A delivered-message or an arriving notification may result in a main-entry and one or more levels of child-entries. The one case defined by this part of ISO/IEC 10021 is when a non-delivery notification contains a returned-content (the **delivered-report entry** is the main-entry and the returned-content is its child-entry, known as a **returned-content entry**).

The rules for how a message-content may be split across several entries is specific to each content-type. A content-specific **synopsis-attribute** may be used to show how the main-entry and the corresponding child-entries are related. When such an attribute is defined, it appears in the International Standard which defines the content-type itself. The **synopsis-attribute** is constructed by the MS.

NOTE - For Interpersonal Messaging (ISO/IEC 10021-7), nested IP-messages within an IP-message are each represented by a child-entry. The ipm-synopsis attribute-type is an example of a content-specific **synopsis-attribute-type**.

An important property of an entry in the stored-messages is its **entry-status**. It is created and maintained by the MS. It can take the following values:

- a) **New** The message has neither been listed by a UA nor has it been automatically processed by the MS.
- b) **Listed** Information about the message has been returned to the UA in either a List abstract-operation or a Fetch abstract-operation, but the message has not yet been completely **processed**.
- c) **Processed** Either a UA has "completely fetched" the message, or the MS has performed some auto-action on it. (Note that some auto-actions result in the message being deleted). The exact definition of "completely fetched" is content-specific and is defined in the corresponding content-specific International Standard.

The **entry-status** of a (non-)delivery-notification becomes **processed** when the delivered-report-envelope is retrieved.

The definition for **entry-status** is as follows;

```
EntryStatus ::= INTEGER (
    new          (0),
    listed       (1),
    processed    (2) }
```

6.5 Auto-actions

6.5.1 Introduction

This subclause defines a framework for the automatic actions (**auto-actions**) which may be registered with the MS.

An **auto-action** is an action that will occur automatically whenever the associated registration criteria have been satisfied. The result of an action being invoked is visible externally to the MS. **Auto-actions** are registered in the MS using the Register-MS abstract-operation (see 8.6).

Each class of **auto-action** is identified by means of an **auto-action-type**. Associated with the registration of an **auto-action**, there is a corresponding **auto-action-registration-parameter**, which are the parameters needed by the MS to perform the registered **auto-action** automatically. The

registration of an **auto-action** requires the use of an **auto-action-registration-identifier** to identify the particular registration.

```
AutoActionRegistration ::= SEQUENCE{
    type                AutoActionType,
    registration-identifier [0] INTEGER (1..ub-per-auto-action) DEFAULT 1,
    registration-parameter [1] ANY DEFINED BY type}
```

6.5.2 Auto-action-type

Some **auto-action-types** will be internationally standardized. Other **auto-action-types** will be defined by national administrative authorities and private organizations. This implies that a number of separate authorities will be responsible for assigning types in a way that ensures that each is distinct from all other assigned **auto-action-types**. This is accomplished by identifying each **auto-action-type** with an object identifier when the **auto-action-type** is defined.

```
AutoActionType ::= OBJECT IDENTIFIER
```

Certain general-purpose **auto-action-types** are defined in clause 12. Such **auto-action-types** are known as **general-auto-action-types** and **auto-actions** of these types as **general-auto-actions**.

6.5.3 Auto-action-registration-parameter

Defining an **auto-action-type** also involves specifying the ASN.1 data-type to which the **auto-action-registration-parameter** shall conform. The data-type of an **registration-parameter** is defined through the object identifier for the **auto-action-type**.

6.5.4 Auto-action-type definition and the AUTO-ACTION Macro

The definition of an **auto-action-type** involves

- a) assigning an object identifier to the **auto-action-type**;
- b) indicating the ASN.1 data-type of the **auto-action-registration-parameter**;

The following ASN.1 macro may (but need not be) used to define an **auto-action-type**

```
AUTO-ACTION MACRO ::=
BEGIN
    TYPE NOTATION      ::= Registration
    VALUE NOTATION     ::= value (VALUE OBJECT IDENTIFIER)

    Registration       ::= "REGISTRATION PARAMETER IS" type
END
```

The correspondence between the parts of the definition, as listed above, and the various pieces of the notation introduced by the **AUTO-ACTION** macro, is as follows:

- a) **Registration**: gives the data-type of the registration parameters associated with an auto-action.
- b) **Value**: the **object identifier** which is used to identify the auto-action.

NOTE - No support is provided in the macro for defining the interaction (if any) between different registrations of the same (or different) **auto-actions**.

6.6 Forwarding of messages

The MS-user makes use of the Message-submission abstract-operation and its parameters as defined in 8.2 of ISO/IEC 10021-4 to request that a message stored in the MS be explicitly forwarded to other users.

The **forwarding-request parameter** is defined using the **EXTENSION** macro defined in clause 9 of ISO/IEC 10021-4 as follows:

```
forwarding-request EXTENSION
  SequenceNumber
  CRITICAL FOR SUBMISSION
  ::= 36
```

If the **SequenceNumber** supplied does not match that of an entry in the **stored-messages** information-base, or matches an entry which is unsuitable for forwarding this is reported using the **Inconsistent-request** abstract-error of 8.2.2.7 of ISO/IEC 10021-4.

7 Abstract-bind and abstract-unbind operations

7.1 Abstract-bind-operation

The **MS-Bind abstract-bind-operation** binds the Indirect-submission, Retrieval and Administration Ports of the MS-user (consumer) to the MS (supplier). The initiator (of the MS-Bind) is the MS-user, while the responder is the MS itself. MS-Bind is defined as follows:

```
MSBind ::= ABSTRACT-BIND
  TO {indirectSubmission[S], retrieval[S], administration[S]}
  BIND
    ARGUMENT MSBindArgument
    RESULT MSBindResult
    BIND-ERROR MSBindError
```

7.1.1 Abstract-bind-argument

The **abstract-bind-argument** parameters are used to identify, authenticate and set the security-context for an MS abstract-service-user. They also contain a set of restrictions for entries to be returned as result of a Fetch abstract-operation, and finally, a request to be informed of the auto-action-types, attribute-types and content-types supported by the MS.

The definition of these parameters is as follows:

```
MSBindArgument ::= SET {
  initiator-name ORAddressAndOrDirectoryName,
  initiator-credentials [2] InitiatorCredentials,
  security-context [3] IMPLICIT SecurityContext OPTIONAL,
  fetch-restrictions [4] Restrictions OPTIONAL -- default is none--,
  ms-configuration-request [5] BOOLEAN DEFAULT FALSE}
```

- 1) **Initiator-name** (C): This argument contains the name of the initiator of the association and is supplied by the initiator. This argument is defined further in 8.1.1.1.1.1 of ISO/IEC 10021-4.
- 2) **Initiator-credentials** (M): This parameter contains the **credentials** of the initiator of the association. It shall be generated by the initiator of the abstract-association.

The **initiator-credentials** may be used by the responder to authenticate the identity of the initiator (see ISO/IEC 9594-8).

If only **simple-authentication** is used, the **initiator-credentials** comprise a simple password.

If **strong-authentication** is used, the **initiator-credentials** comprise an **initiator-bind-token**, and, optionally, an **initiator-certificate**. The **initiator-bind-token** and **initiator-certificate** are defined further in 8.1.1.1.1.2 of ISO/IEC 10021-4. The **initiator-credentials** of the MS-user may differ from the **initiator-credentials** used in the **MTS-bind** as defined in 8.1.1.1.1.2 of ISO/IEC 10021-4.

- 3) **Security-context** (O): This parameter identifies the **security-context** that the initiator of the abstract-association proposes to operate at. It is generated by the initiator of the abstract-association. The **security-context** is defined further in 8.1.1.1.1.3 of ISO/IEC 10021-4.

The **security-context** comprises one or more **security-labels** that define the sensitivity of interactions that may occur between the MS abstract-service-user and the MS abstract-service for the duration of the abstract-association, in line with the **security-policy** in force. The **security-context** shall be one that is allowed by the registered **user-security-labels** of the MS abstract-service-user and by the **security-labels** associated with the MS.

In the absence of this parameter, **security-contexts** are not established between the MS abstract-service-user and the MS abstract-service, and the sensitivity of interactions that may occur between the MS abstract-service-user and the MS abstract-service is at the discretion of the invoker of the abstract-service.

- 4) **Fetch-restrictions** (O): This contains the restrictions on entries to be returned as result of a Fetch abstract-operation. The **fetch-restrictions** remain set until an abstract-unbind-operation is issued.

In the absence of this argument, the default is that no fetch-restrictions need to be performed.

This argument consists of the following components:

```
Restrictions ::= SET {
    allowed-content-types      [0] SET SIZE (1 .. ub-content-types) OF OBJECT IDENTIFIER OPTIONAL
    --default is no restriction--,
    allowed-EITs              [1] MS-EITs OPTIONAL --default is no restriction--,
    maximum-content-length    [2] ContentLength OPTIONAL --default is no restriction--}
```

- a) **Allowed-content-types** (C): The content-types that the MS abstract-service-user is prepared to accept as result of a Fetch abstract-operation. Any message with a content-type other than the ones specified will not be returned, but result in an error, unless the Fetch abstract-operation has explicitly overridden the restriction.

In the absence of this component, the default is that no fetch-restrictions on content-types need to be performed.

- b) **Allowed-EITs** (C): The encoded-information-types that the MS abstract-service-user is prepared to accept as result of a Fetch abstract-operation. If a message contains encoded-information-types other than the ones specified, a filtering will take place so that disallowed EIT parts are not returned along with the text of the message. If the whole message consists of disallowed EITs, an error will be reported. No filtering will take place if the Fetch abstract-operation has explicitly overridden the restriction.

```
MS-EITs ::= SET SIZE (1..ub-encoded-information-types) OF MS-EIT
```

```
MS-EIT ::= OBJECT IDENTIFIER
```

In the absence of this component, the default is that no fetch-restrictions on encoded-information-types need to be performed.

- c) **Maximum-content-length** (C): The maximum content length that the MS abstract-service-user is prepared to accept as result of a Fetch abstract-operation. Any message with a **content-length** exceeding the one specified will not be returned, but result in an error, unless the Fetch abstract-operation has explicitly overridden the restriction.

In the absence of this component, the default is that no fetch-restrictions on content-length need to be performed.

- 5) **MS-configuration-request (C)**: The **MS-configuration-request** is requested to obtain information relating to which auto-actions and optional attributes the MS provides support for.

In the absence of this component, the default is false which indicates that no such request is being made.

7.1.2 Abstract-bind-result

The **abstract-bind-result** parameters are as follows:

MSBindResult ::= SET {	
responder-credentials	[2] ResponderCredentials,
available-auto-actions	[3] SET SIZE (1 .. ub-auto-actions) OF AutoActionType OPTIONAL,
available-attribute-types	[4] SET SIZE (1 .. ub-attributes-supported) OF AttributeType OPTIONAL,
alert-indication	[5] BOOLEAN DEFAULT FALSE,
content-types-supported	[6] SET SIZE (1..ub-content-types) OF OBJECT IDENTIFIER OPTIONAL}

- 1) **Responder-credentials (M)**: This parameter contains the credentials of the responder of the abstract-association. It shall be generated by the responder of the abstract-association.

The **responder-credentials** may be used by the initiator to authenticate the identity of the responder (see ISO/IEC 9594-8).

If only **simple-authentication** is used, the **responder-credentials** comprise a simple **password** associated with the responder.

If **strong-authentication** is used, the **responder-credentials** comprise a **responder-bind-token**, and, optionally, a **responder-certificate**, both of which are generated by the responder of the abstract-association. The **responder-bind-token** and **responder-certificate** are defined further in 8.1.1.1.2.2 of ISO/IEC 10021-4.

- 2) **Available-auto-actions (C)**: Specifies the set of all possible auto-actions that are supported by the MS (not just those requested by the MS abstract-service-user). Only present if an **MS-configuration-request** is made.
- 3) **Available-attribute-types (C)**: Specifies the set of all optional attributes supported by the MS. Only present if an **MS-configuration-request** is made.
- 4) **Alert-indication (C)**: If true, then an alert condition has occurred since the last successful **Alert-indication**.
- 5) **Content-types-supported (C)**: Specifies a set of object-identifiers that define the content-types that the MS has knowledge of. Only present if an **MS-configuration-request** is made.

7.1.3 Abstract-bind-errors

There are two possible errors defined by the **Retrieval Port**, namely **authentication-error** and **unacceptable-security-context**.

The definition of the errors is

```

MSBindError :: = ENUMERATED {
    authentication-error      (0),
    unacceptable-security-context (1),
    unable-to-establish-association (2)}

```

- 1) **Authentication-error (C)**: This error reports that an abstract-association cannot be established because the initiator's **credentials** are not acceptable or are improperly specified.

The **authentication-error** has no parameters.

- 2) **Unacceptable-security-context (C)**: This error reports that the **security-context** proposed by the initiator of the abstract-association is unacceptable to the responder.

The **unacceptable-security-context** error has no parameters.

- 3) **Unable-to-establish-association (C)**: This error reports that the responder has rejected the initiator's attempt to establish an abstract-association.

The **unable-to-establish-association** error has no parameters.

7.2 Abstract-unbind-operation

The **MS-Unbind** abstract-unbind-operation closes the abstract-association. The issuing of an abstract-unbind-operation results in the relaxation of any fetch-restrictions that were specified in the abstract-bind operation argument. There is not argument, result, or error associated with the abstract-unbind-operation.

```

MSUnbind :: = ABSTRACT-UNBIND
    FROM {indirectSubmission[S], retrieval[S], administration[S]}

```

8 Abstract-operations

This clause defines the following **abstract-operations** available at the Retrieval Port

- a) Summarize;
- b) List;
- c) Fetch;
- d) Delete;
- e) Register-MS;
- f) Alert.

The MS is the MS abstract-service-provider of each of these **abstract-operations**. For the formal definition of the Retrieval Port see 6.2.

The abstract-operations may be performed asynchronously subject to the following conditions. The Delete and Register-MS abstract-operations shall not be performed until all outstanding abstract-operations have been completed. Additionally these abstract-operations are performed in the order in which they are invoked and are required to complete prior to any other abstract-operations being performed. As a consequence of this and the fact that the List and Fetch abstract-operations change the status of a message entry, the results of the Summarize, List and Fetch abstract-operations may be non-deterministic.

8.1 Common Data-types used in abstract-operations

This subclause defines a number of common data-types which are used in several of the **abstract-operations** defined in the remainder of clause 8. Many of the **abstract-operations** also make use of entries and attributes as defined in 6.3.

The common data-types defined in this part of ISO/IEC 10021 are

- a) Range;
- b) Filter;
- c) Selector;
- d) Entry Information Selection;
- e) Entry Information.

8.1.1 Range

A **range** parameter is used to select a contiguous sequence of entries from an information-base.

```
Range ::= CHOICE {  
    sequence-number-range    [0] NumberRange ,  
    creation-time-range      [1] TimeRange }
```

```

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

```

```

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

```

```

CreationTime ::= UTCTime

```

The components of **range** have the following meanings:

1) **Sequence-number-range** (C), and

2) **Creation-time-range** (C): Both of these parameters identify the contiguous sequence of entries to be selected. The **sequence-number-range** is given in terms of **sequence-numbers**, and the **creation-time-range** is given in terms of **creation-times**. The **creation-time** of an entry is the time at which the MS generated the entry. The **sequence-numbers** of successive entries are always in ascending order, but several adjacent entries may have the same **creation-time**. The parameters of both **number-range** and **time-range** have the following meanings:

a) **From** (O): This is the lower bound for the **range**.

In the absence of this component, the default is **no lower bound**, and the selection starts with the earliest message (lowest **sequence-number**) in the information-base.

b) **To** (O): This is the upper bound for the **range**.

In the absence of this component, the default is **no upper bound**, and the selection finishes with the latest message (highest **sequence-number**) in the information-base.

8.1.2 Filters

8.1.2.1 Filter

A **filter** parameter applies a test to a particular entry and is either satisfied or not by the entry. The **filter** is expressed in terms of assertions about the presence or value of certain attributes of the entry, and is satisfied if and only if it evaluates to **true**.

```

Filter ::= CHOICE {
    item      [0] FilterItem,
    and       [1] SET SIZE (1 .. ub-nested-filters) OF Filter,
    or        [2] SET SIZE (1 .. ub-nested-filters) OF Filter,
    not       [3] Filter}

```

A **filter** is either a **filter-item**, or an expression involving simpler **filters** composed together using the logical operators **and**, **or**, and **not**.

Where the **filter** is

- a) an **item**, it is **true** if and only if the corresponding **filter-item** is **true**;
- b) an **and**, it is **true** unless any of the **filters** in the **SET** are **false**;

NOTE- Thus, if there are no **filters** in the **SET**, the **and** evaluates to **true**.

c) an **or**, it is **false** unless any of the **filters** in the **SET** are **true**;

NOTE - Thus, if there are no **filters** in the **SET**, the **or** evaluates to **false**.

d) a **not**, it is **true** if and only if the **filter** is **false**.

8.1.2.2 Filter-item

A **filter-item** is an assertion about the presence or value(s) of an attribute of a particular type in the entry under test. Each such assertion is either **true** or **false**.

FilterItem ::= CHOICE {	
equality	[0] AttributeValueAssertion,
substrings	[1] SEQUENCE {
type	AttributeType,
strings	SEQUENCE SIZE (1.. ub-attribute-values) OF CHOICE {
initial	[0] ANY --DEFINED BY type--,
any	[1] ANY --DEFINED BY type--,
final	[2] ANY --DEFINED BY type--}},
greater-or-equal	[2] AttributeValueAssertion ,
less-or-equal	[3] AttributeValueAssertion,
present	[4] AttributeType,
approximate-match	[5] AttributeValueAssertion}

Every filter-item includes an attribute-type which identifies the particular attribute concerned.

Any assertion about the value of such an attribute is only evaluated if the attribute-type is defined, and the purported attribute-value(s) are of the data-type defined for attribute-values of that attribute.

Assertions about the value of an attribute are evaluated by matching the attribute for EQUALITY, SUBSTRINGS, and ORDERING, as defined in 6.3.3.4.

Where the **filter-item** asserts

- a) **equality**, it is **true** if and only if there is a value of the attribute which is equal to that asserted;
- b) **substrings**, it is **true** if and only if there is a value of the attribute in which the specified **substrings** appear in the given order. The **substrings** shall be non-overlapping, and may (but need not) be separated from the ends of the attribute-value and from one another by zero or more **string** elements.

The first character in **initial**, if present, shall match the first character in the attribute-value; the last character in **final**, if present, shall match the last character in the attribute-value; **any**, if present, may match any substring in the attribute-value;

- c) **greater-or-equal**, it is **true** if and only if the relative ordering places the supplied value after any value of the attribute;
- d) **less-or-equal**, it is **true** if and only if the relative ordering places the supplied value before any value of the attribute;
- e) **present**, it is **true** if and only if such an attribute is present in the entry;