

# INTERNATIONAL STANDARD

**ISO/IEC**  
**14363**

First edition  
1996-06-01

---

---

## **Information technology — Test methods for measuring conformance to MHS-based electronic messaging — Application Program Interface (API) [Language independent]**

*Technologies de l'information — Méthodes d'essai pour mesurer la  
conformité à la messagerie électronique basée sur MHS — Interface de  
programme d'application (API) [indépendante du langage]*



Reference number  
ISO/IEC 14363:1996(E)  
**AVA Documents**  
Document Sharing Hub

## Contents

|                                                                               | PAGE |
|-------------------------------------------------------------------------------|------|
| Section 1: General . . . . .                                                  | 1    |
| 1.1 Scope . . . . .                                                           | 1    |
| 1.2 Normative References . . . . .                                            | 1    |
| 1.3 Conformance . . . . .                                                     | 2    |
| Section 2: Terminology and General Requirements . . . . .                     | 3    |
| 2.1 Conventions . . . . .                                                     | 3    |
| 2.2 Definitions . . . . .                                                     | 4    |
| Section 3: Test Assertions for Section General . . . . .                      | 7    |
| 3.1 Scope . . . . .                                                           | 7    |
| 3.2 Normative References . . . . .                                            | 7    |
| 3.3 Conformance . . . . .                                                     | 7    |
| 3.3.1 Implementation Conformance . . . . .                                    | 7    |
| 3.3.2 Application Conformance . . . . .                                       | 10   |
| 3.4 Test Assertions . . . . .                                                 | 10   |
| Section 4: Test Assertions for Terminology and General Requirements . . . . . | 11   |
| Section 5: Test Assertions for Overview . . . . .                             | 13   |
| 5.1 Purpose . . . . .                                                         | 13   |
| 5.2 Overview . . . . .                                                        | 13   |
| 5.3 Object Management . . . . .                                               | 13   |
| 5.4 Features . . . . .                                                        | 13   |
| Section 6: Test Assertions for Message Handling Interfaces . . . . .          | 15   |
| 6.1 Datatypes . . . . .                                                       | 15   |
| 6.1.1 Boolean List . . . . .                                                  | 15   |
| 6.1.2 Feature List . . . . .                                                  | 15   |
| 6.1.3 Interval . . . . .                                                      | 15   |
| 6.1.4 Object Count . . . . .                                                  | 16   |
| 6.1.5 Sequence Number . . . . .                                               | 16   |
| 6.2 Access Operations . . . . .                                               | 16   |
| 6.2.1 Cancel Submission . . . . .                                             | 19   |
| 6.2.2 Close . . . . .                                                         | 22   |
| 6.2.3 Finish Delivery . . . . .                                               | 24   |
| 6.2.4 Finish Retrieval . . . . .                                              | 28   |
| 6.2.5 Open . . . . .                                                          | 30   |
| 6.2.6 Size . . . . .                                                          | 35   |

© ISO/IEC 1996

All rights reserved. Unless otherwise specified, 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.

ISO/IEC Copyright Office • Case postale 56 • CH-1211 Genève 20 • Switzerland

Printed in Switzerland

|            |                                                                |     |
|------------|----------------------------------------------------------------|-----|
| 6.2.7      | Start Delivery . . . . .                                       | 38  |
| 6.2.8      | Start Retrieval . . . . .                                      | 113 |
| 6.2.9      | Submit . . . . .                                               | 189 |
| 6.2.10     | Wait . . . . .                                                 | 227 |
| 6.3        | Transfer Operations . . . . .                                  | 231 |
| 6.3.1      | Close . . . . .                                                | 232 |
| 6.3.2      | Finish Transfer In . . . . .                                   | 234 |
| 6.3.3      | Open . . . . .                                                 | 236 |
| 6.3.4      | Size . . . . .                                                 | 241 |
| 6.3.5      | Start Transfer In . . . . .                                    | 243 |
| 6.3.6      | Transfer Out . . . . .                                         | 319 |
| 6.3.7      | Wait . . . . .                                                 | 350 |
| 6.4        | Return Codes . . . . .                                         | 352 |
| Section 7: | Test Assertions for Interpersonal Messaging Packages . . . . . | 359 |
| 7.1        | Class Hierarchy . . . . .                                      | 359 |
| 7.2        | Class Definitions . . . . .                                    | 359 |
| 7.3        | Class Definitions . . . . .                                    | 360 |
| 7.3.1      | Bilaterally Defined Body Part . . . . .                        | 360 |
| 7.3.2      | Body Part . . . . .                                            | 361 |
| 7.3.3      | Externally Defined Body Part . . . . .                         | 361 |
| 7.3.4      | G3 Fax Body Part . . . . .                                     | 362 |
| 7.3.5      | G4 Class 1 Body Part . . . . .                                 | 364 |
| 7.3.6      | General Text Body Part . . . . .                               | 364 |
| 7.3.7      | IA5 Text Body Part . . . . .                                   | 366 |
| 7.3.8      | Interpersonal Message . . . . .                                | 367 |
| 7.3.9      | Interpersonal Notification . . . . .                           | 381 |
| 7.3.10     | IPM Identifier . . . . .                                       | 384 |
| 7.3.11     | IPMS Extension . . . . .                                       | 386 |
| 7.3.12     | Message Body Part . . . . .                                    | 386 |
| 7.3.13     | Mixed-mode Body Part . . . . .                                 | 388 |
| 7.3.14     | Nationally Defined Body Part . . . . .                         | 389 |
| 7.3.15     | Non-receipt Notification . . . . .                             | 389 |
| 7.3.16     | Office Document Architecture Body Part . . . . .               | 392 |
| 7.3.17     | OR Descriptor . . . . .                                        | 394 |
| 7.3.18     | Receipt Notification . . . . .                                 | 396 |
| 7.3.19     | Recipient Specifier . . . . .                                  | 398 |
| 7.3.20     | Teletex Body Part . . . . .                                    | 401 |
| 7.3.21     | Unidentified Body Part . . . . .                               | 403 |
| 7.3.22     | Videotex Body Part . . . . .                                   | 405 |
| 7.4        | Syntax Definitions . . . . .                                   | 407 |
| 7.4.1      | Acknowledgement . . . . .                                      | 407 |
| 7.4.2      | Auto-submitted . . . . .                                       | 407 |
| 7.4.3      | Discard Reason . . . . .                                       | 408 |
| 7.4.4      | IA5 Repertoire . . . . .                                       | 410 |
| 7.4.5      | Importance . . . . .                                           | 411 |
| 7.4.6      | Non-receipt Reason . . . . .                                   | 411 |
| 7.4.7      | Notification Request . . . . .                                 | 412 |
| 7.4.8      | Sensitivity . . . . .                                          | 413 |
| 7.4.9      | Videotex Syntax . . . . .                                      | 414 |

|                                                                        |     |
|------------------------------------------------------------------------|-----|
| Section 8: Test Assertions for Section Message Handling Packages . . . | 417 |
| 8.1 Class Hierarchy . . . . .                                          | 417 |
| 8.2 Class Definitions . . . . .                                        | 418 |
| 8.2.1 Algorithm . . . . .                                              | 418 |
| 8.2.2 Algorithm and Result . . . . .                                   | 419 |
| 8.2.3 Asymmetric Token . . . . .                                       | 419 |
| 8.2.4 Bilateral Information . . . . .                                  | 421 |
| 8.2.5 Communique . . . . .                                             | 423 |
| 8.2.6 Content . . . . .                                                | 432 |
| 8.2.7 Delivered Message . . . . .                                      | 432 |
| 8.2.8 Delivered Per-recipient DR . . . . .                             | 432 |
| 8.2.9 Delivered Per-recipient NDR . . . . .                            | 434 |
| 8.2.10 Delivered Per-recipient Report . . . . .                        | 436 |
| 8.2.11 Delivered Report . . . . .                                      | 439 |
| 8.2.12 Delivery Confirmation . . . . .                                 | 440 |
| 8.2.13 Delivery Envelope . . . . .                                     | 441 |
| 8.2.14 Delivery Report . . . . .                                       | 452 |
| 8.2.15 EITs . . . . .                                                  | 459 |
| 8.2.16 Expansion Record . . . . .                                      | 460 |
| 8.2.17 Extensible Object . . . . .                                     | 461 |
| 8.2.18 Extension . . . . .                                             | 463 |
| 8.2.19 External Trace Entry . . . . .                                  | 465 |
| 8.2.20 G3 Fax NBPs . . . . .                                           | 469 |
| 8.2.21 General Content . . . . .                                       | 471 |
| 8.2.22 Internal Trace Entry . . . . .                                  | 471 |
| 8.2.23 Local Delivery Confirmation . . . . .                           | 477 |
| 8.2.24 Local Delivery Confirmations . . . . .                          | 477 |
| 8.2.25 Local NDR . . . . .                                             | 478 |
| 8.2.26 Local Per-recipient NDR . . . . .                               | 478 |
| 8.2.27 Message . . . . .                                               | 480 |
| 8.2.28 Message RD . . . . .                                            | 484 |
| 8.2.29 MT Public Data . . . . .                                        | 487 |
| 8.2.30 MTS Identifier . . . . .                                        | 489 |
| 8.2.31 OR Address . . . . .                                            | 491 |
| 8.2.32 OR Name . . . . .                                               | 494 |
| 8.2.33 Per-recipient DR . . . . .                                      | 496 |
| 8.2.34 Per-recipient NDR . . . . .                                     | 498 |
| 8.2.35 Per-recipient Report . . . . .                                  | 500 |
| 8.2.36 Probe . . . . .                                                 | 502 |
| 8.2.37 Probe RD . . . . .                                              | 504 |
| 8.2.38 RD . . . . .                                                    | 507 |
| 8.2.39 Redirection Record . . . . .                                    | 507 |
| 8.2.40 Report . . . . .                                                | 509 |
| 8.2.41 Security Label . . . . .                                        | 512 |
| 8.2.42 Session . . . . .                                               | 514 |
| 8.2.43 Submission Results . . . . .                                    | 516 |
| 8.2.44 Submitted Communique . . . . .                                  | 518 |
| 8.2.45 Submitted Message . . . . .                                     | 521 |
| 8.2.46 Submitted Message RD . . . . .                                  | 526 |
| 8.2.47 Submitted Probe . . . . .                                       | 529 |

|                          |                                                          |     |
|--------------------------|----------------------------------------------------------|-----|
| 8.2.48                   | Submitted Probe RD                                       | 531 |
| 8.2.49                   | Teletex NBPs                                             | 534 |
| 8.2.50                   | Token                                                    | 536 |
| 8.2.51                   | Token Public Data                                        | 536 |
| 8.3                      | Syntax Definitions                                       | 536 |
| 8.3.1                    | Action                                                   | 536 |
| 8.3.2                    | Builtin EIT                                              | 537 |
| 8.3.3                    | Delivery Mode                                            | 538 |
| 8.3.4                    | Delivery Point                                           | 539 |
| 8.3.5                    | Diagnostic                                               | 541 |
| 8.3.6                    | Explicit Conversion                                      | 549 |
| 8.3.7                    | Postal Mode                                              | 550 |
| 8.3.8                    | Postal Report                                            | 551 |
| 8.3.9                    | Priority                                                 | 552 |
| 8.3.10                   | Reason                                                   | 552 |
| 8.3.11                   | Redirection Reason                                       | 554 |
| 8.3.12                   | Registration                                             | 555 |
| 8.3.13                   | Report Request                                           | 556 |
| 8.3.14                   | Security Classification                                  | 557 |
| 8.3.15                   | Terminal Type                                            | 557 |
| Section 9:               | Test Assertions for Section Secure Messaging Package     | 559 |
| 9.1                      | Class Hierarchy                                          | 560 |
| 9.2                      | Class Definitions                                        | 560 |
| 9.2.1                    | Integrity Check Basis                                    | 561 |
| 9.2.2                    | Origin Check Basis                                       | 561 |
| 9.2.3                    | Per-recipient Check Basis                                | 563 |
| 9.2.4                    | Per-recipient Delivery Check Basis                       | 563 |
| 9.2.5                    | Per-recipient Non-delivery Check Basis                   | 565 |
| 9.2.6                    | Proof of Delivery Basis                                  | 565 |
| 9.2.7                    | Proof of Submission Basis                                | 566 |
| 9.2.8                    | MT Secret Data                                           | 567 |
| 9.2.9                    | Token Secret Data                                        | 568 |
| 9.3                      | Syntax Definitions                                       | 569 |
| Section 10:              | Test Assertions for Section Definition of Constants      | 571 |
| 10.1                     | Object Identifier for This International Standard        | 571 |
| 10.2                     | OIDs for Features and Packages                           | 571 |
| 10.3                     | Interpersonal Messaging Package OIDs                     | 573 |
| 10.4                     | Message Handling Package Value Definitions               | 575 |
| 10.5                     | Secure Messaging Package Value Definitions               | 578 |
| Annex A (informative)    | Test Assertions for Annex Functional Architecture of API | 579 |
| A.1                      | Message Access Interface                                 | 579 |
| A.2                      | Message Transfer Interface                               | 581 |
| Annex B (informative)    | Bibliography                                             | 583 |
| Alphabetic Topical Index |                                                          | 585 |

## 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 14363 was prepared by IEEE (as IEEE Std 1326.1-1993) and was adopted, under a special “fast-track procedure”, by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, in parallel with its approval by national bodies of ISO and IEC.

Annexes A and B of this International Standard are for information only.

## 1 Introduction

(This introduction is not a normative part of ISO/IEC 14363, Information technology—Test methods for measuring conformance to MHS-based electronic messaging—Application Program Interface (API) [Language independent], but is included for information only.)

The purpose of this International Standard is to define test methods for the language-independent MHS-based electronic messaging API contained in ISO/IEC 14361 {3}.

## 8 Related Standards

ISO/IEC 14361 {3} is intended to provide the basis for the definition of programming language bindings to which implementations and applications can conform. A specification for such a language binding, for the C programming language, is contained in ISO/IEC 14365 {B2}. This International Standard applies to test methods for measuring conformance to any such programming language binding specification.

The API defined in ISO/IEC 14361 {3} uses the mechanism for OSI abstract data manipulation (OM) defined in ISO/IEC 14360 {2}. ISO/IEC 14362 {4} defines the requirements that shall apply to test methods for measuring conformance to ISO/IEC 14360 {2}. A set of test methods used to measure conformance to ISO/IEC 14361 {3} shall satisfy the requirements of ISO/IEC 14362 {4}, as well as conforming to this International Standard.

IEEE Std 1003.3-1991 {5} defines the general requirements that shall apply to test methods for measuring conformance to POSIX. A set of test methods used to measure conformance to ISO/IEC 14361 {3} shall satisfy the requirements of IEEE Std 1003.3-1991 {5}, as well as conforming to this International Standard.

## 25 Overview

For each section of ISO/IEC 14361 {3}, this International Standard contains a corresponding section containing test assertions, in accordance with IEEE Std 1003.3-1991 {5}. A set of test methods conforms to this International Standard if it tests all of the test assertions.

## Related Standards Activities

The following areas are under active consideration at this time, or are expected to become active in the near future, concerning extensions to this International Standard and similar efforts can be anticipated in the future:<sup>1)</sup>

- (1) Directory services
- (2) FTAM API
- (3) Verification testing methods
- (4) Network interface facilities
- (5) System administration.

This International Standard is based on IEEE Std 1326.1-1993 (B4), which was prepared by the P1224 Working Group, sponsored by the Portable Applications Standards Committee of the IEEE Computer Society.

---

1) A *Standards Status Report* that lists all current IEEE Computer Society standards projects is available from the IEEE Computer Society, 1730 Massachusetts Avenue NW, Washington, DC 20036-1903; Telephone: +1 202 371-0101; FAX: +1 202 728-9614.

# Information technology—Test methods for measuring conformance to MHS-based electronic messaging—Application Program Interface (API) [Language independent]

## Section 1: General

### 1.1 Scope

This International Standard defines requirements for test methods for measuring conformance to ISO/IEC 14361 {3}.

ISO/IEC 14361 {3} is stated in terms that are independent of any particular programming language, but each implementation that conforms to it implements a particular programming language binding, and conforms to its programming language binding specification. Each set of test methods for measuring conformance to ISO/IEC 14361 {3} assumes a particular programming language binding specification. Each programming language binding specification may impose further programming language specific requirements on the test methods which, in conjunction with the requirements imposed by this International Standard, constitute the requirements that shall be satisfied by test methods used for measuring conformance to that programming language binding specification. This International Standard applies to all sets of test methods for measuring conformance to any programming language binding specification for ISO/IEC 14361 {3}.

### 1.2 Normative References

The following standards contain provisions which, through reference in this text, constitute provisions of this International Standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this International Standard are encouraged to investigate the possibility of applying the most recent editions of the standards indicated

below. Members of IEC and ISO maintain registers of currently valid International Standards.

- {1} ISO 3166: 1993,<sup>1)</sup> *Codes for the Representation of Names of Countries*.
- {2} ISO/IEC 14360: 1996, *Information technology—Open Systems Interconnection (OSI) abstract data manipulation—Application Program Interface (API) [Language independent]*.
- {3} ISO/IEC 14361: 1996, *Information technology—MHS-based electronic messaging—Application Program Interface (API) [Language independent]*.
- {4} ISO/IEC 14362: 1996, *Information technology—Test methods for measuring conformance to Open Systems Interconnection (OSI) abstract data manipulation—Application Program Interface (API) [Language independent]*.
- {5} IEEE Std 1003.3-1991,<sup>2)</sup> *IEEE Standard for Information Technology—Test Methods for Measuring Conformance to POSIX*.
- {6} CCITT Recommendation X.121: 1984,<sup>3)</sup> *International Numbering Plan for Public Data Networks. CCITT Red Book, Fascicle VIII.4*.

### 1.3 Conformance

A set of test methods that conforms to this International Standard shall conform to IEEE Std 1003.3-1991 {5}, with references to the “POSIX.n test method specification” being interpreted as references to this International Standard, and references to “the POSIX standard for which conformance is being measured” being interpreted as references to ISO/IEC 14361 {3}.

In addition to meeting the conformance criteria defined in IEEE Std 1003.3-1991 {5}, a set of test methods that conforms to this International Standard shall test all documentation assertions defined in this International Standard.

NOTE: Conformance to IEEE Std 1003.3 {5} implies that the test methods will test all other assertions defined in this International Standard.

- 
- 1) ISO/IEC documents can be obtained from the ISO Central Secretariat, 1 Rue de Varembé, Case Postale 56, CH-1211, Genève 20, Switzerland/Suisse.
  - 2) IEEE documents can be obtained from the IEEE Service Center, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ, 08855-1331, USA. Telephone: 1(800)678-IEEE or +1 (908) 981-1393 (outside US).
  - 3) CCITT documents can be obtained from the Telecommunication Standardization Bureau of the International Telecommunication Union, Sales Section, Place des Nations, CH-1211, Genève 20, Switzerland/Suisse.

## Section 2: Terminology and General Requirements

### 2.1 Conventions

#### 2.1.1 General and Typographical Conventions

Language-independent concrete OM class names, OM attribute names, and OM-attribute value names are spelled with hyphens between words (e.g., **conversion-loss-prohibited**). The first letters of language-independent OM class and OM attribute names are capitalized (e.g., **Redirection-Record**).

Language-independent datatype, operation, argument and error names are lowercase and are spelled with underscores between words (e.g., `ma_start_delivery`).

The use of fonts in this International Standard is as follows:

- The Helvetica font is used for:
  - Language-independent operation names, such as `ma_start_delivery`
  - Language-independent datatype names, such as `mh_interval_type`
  - Language-independent error names, such as `feature_unavailable`
- The *italic* font is used for:
  - Language-independent operation arguments, such as *user\_address*
  - The introduction of important terms
  - Cross-references in 2.2
- The **bold** font is used for:
  - Language-independent concrete OM class names, such as **Delivered-Report**
  - Language-independent OM attribute names, such as **Algorithm-Datum**
  - Language-independent OM attribute values, such as **conversion-loss-prohibited**
- The ***bold italic*** font is used for:
  - Language-independent abstract OM class names, such as ***Delivery-Report***

— The constant width (Courier) font is used for:

- 28 — References to terms defined in international standards upon which this  
29 International Standard is based.

## 30 2.2 Definitions

### 31 2.2.1 Terminology

32 For the purposes of this International Standard, the following definitions apply:

33 **2.2.1.1 conformance document:** A document provided by an implementor that  
34 contains implementation details. [ISO/IEC 9945-1 {B1}]

35 **2.2.1.2 may:** An indication of an optional feature. [ISO/IEC 9945-1 {B1}]

36 With respect to implementations, the word *may* is to be interpreted as an optional  
37 feature that is not required in this International Standard, but that can be pro-  
38 vided. With respect to Strictly Conforming Applications, the word *may* means that  
39 the optional feature shall not be used.

40 **2.2.1.3 shall:** An indication of a requirement on the implementation or on Strictly  
41 Conforming Applications, where appropriate. [ISO/IEC 9945-1 {B1}]

42 **2.2.1.4 should:**

- 43 (1) With respect to implementations, an indication of an implementation  
44 recommendation, but not a requirement.
- 45 (2) With respect to applications, an indication of a recommended program-  
46 ming practice for applications and a requirement for Strictly Conforming  
47 Applications. [ISO/IEC 9945-1 {B1}]

48 **2.2.1.5 supported:** A condition regarding optional functionality.

49 Certain functionality in this International Standard is optional, but the interfaces  
50 to that functionality are always required. If the functionality is *supported*, the  
51 interfaces work as specified by this International Standard (except that they do not  
52 return the error condition indicated for the unsupported case). If the functionality  
53 is not *supported*, the interface shall always return the indication specified for this  
54 situation. [ISO/IEC 9945-1 {B1}]

55 **2.2.1.6 system documentation:** All documentation provided with an implemen-  
56 tation, except the conformance document.

57 Electronically distributed documents for an implementation are considered part of  
58 the system documentation. [ISO/IEC 9945-1 {B1}]

## 2.2.2 General Terms

59 For the purposes of this International Standard, the following definitions apply:

60 **2.2.2.1 assertion:** A statement that is derived from the standard to which confor-  
61 mance is being measured, that is true for a conforming implementation, and that  
62 pertains either to functionality or behavior of a functional interface or namespace  
63 allocation or to the documentation associated with the implementation being tested  
64 [ISO/IEC 14360 {2}].

65 **2.2.2.2 generic interface:** A version of an interface that is independent of any  
66 particular programming language.

67 **2.2.2.3 programming language binding specification:** For a language-  
68 independent specification, a document that specifies, in terms of a particular pro-  
69 gramming language, the behavior that the language-independent specification  
70 specifies in language-independent terms.

71 It may also specify additional behavior that is relevant to the usage of the particu-  
72 lar programming language [ISO/IEC 14360 {2}].

73 **2.2.2.4 test methods:** The software, procedures, or other means specified to  
74 measure conformance to a specification [ISO/IEC 14360 {2}].

## 75 2.2.3 Abbreviations

76 For the purposes of this International Standard, the following abbreviations apply:

77 **2.2.3.1 OM:** Object Management.

## 1                    **Section 3: Test Assertions for Section General**

### 2           **3.1 Scope**

3           There are no test assertions for this clause.

### 4           **3.2 Normative References**

5           There are no test assertions for this clause.

### 6           **3.3 Conformance**

#### 7           **3.3.1 Implementation Conformance**

##### 8           **3.3.1.1 Conformance Requirements**

9           D01           The manufacturer shall identify the interface (MA, MT, or both) that  
10                     the product implements, and state what roles (client, service, or both)  
11                     it plays for each.

12           001(A)       The product shall implement the OM interface as defined in ISO/IEC  
13                     14360 (2), satisfying its conformance requirements, and play the same  
14                     roles for that interface as it plays for the MA and MT interfaces.

15           D02           If the product plays the role of service for the MA or MT interface:  
16                     The manufacturer shall state which features the product imple-  
17                     ments.

18           002(C)       If the product plays the role of service for the MA interface:  
19                     The product shall implement the Basic Access FU.

20           003(C)       If the product plays the role of service for the MA interface:  
21                     The product shall implement the Submission FU, the Delivery  
22                     FU, the Retrieval FU, or any combination of the three.

23           004(C)       If the product plays the role of service for the MT interface:  
24                     The product shall implement the Basic Transfer FU.

- 25 005(C) If the product plays the role of service for the MT interface:  
 26 The product shall implement the Transfer In FU, the Transfer  
 Out FU, or both.
- 27 006(C) If the product plays the role of service for the MA or MT interface:  
 28 The product shall implement the MH 88 Package, the MH 84  
 29 Package, or both.
- 30 GA01 For *FU* of Basic Access FU, Submission FU, Delivery FU, Retrieval  
 31 FU, Basic Transfer FU, Transfer In FU, Transfer Out FU:  
 32 If the product plays the role of service for FU:  
 33 The product shall implement every aspect of every func-  
 34 tion in FU.
- 35 D03 For *package* of IM 88 Package, MH 88 Package:  
 36 If the product implements *package*:  
 37 The product documentation shall state which (if any) of  
 38 the 1988 classes of *package* the product implements.
- 39 D04 For *package* of IM 88 Package, IM 84 Package:  
 40 If the product implements *package*:  
 41 The product documentation shall state which (if any)  
 42 immediate subclasses of the **Body-Part** class the pro-  
 43 duct implements.
- 44 GA02 For *package* of IM 84 Package, SM 88 Package:  
 45 If the product implements *package*:  
 46 The product shall implement every class the *package*  
 47 contains.
- 48 007(C) If the product implements the MH 84 package:  
 49 The product shall implement all of those classes that are used  
 50 by the functional units that it supports.
- 51 008(A) The product shall implement the closures of all classes it implements.
- 52 D05 The product documentation shall state what classes it provides the  
 53 om\_encode function for.
- 54 D06 If the product plays the role of service for the MA interface:  
 55 The manufacturer shall state whether or not the product real-  
 56 izes the interface by means of X.400's *Message Submission and*  
 57 *Delivery Protocol (P3)* or *Message Retrieval Protocol (P7)*.
- 58 D07 If the product plays the role of service for the MT interface:  
 59 The manufacturer shall state whether or not it realizes the  
 60 interface by means of X.400's *Message Transfer Protocol (P1)*.
- 61 D08 If the product plays the role of service for the MA interface or the MT  
 62 interface and implements the IM 88 Package or the IM 84 Package:  
 63 The manufacturer shall state whether or not the product real-  
 64 izes the interface by means of X.400's *Interpersonal Messaging*  
 65 *Protocol (P2)*.
- 66 GA03 For *protocol* of P1 (1988), P3 (1988), P7 (1988):  
 67 If the product implements *protocol*:  
 The manufacturer and product shall satisfy the

- 68 conformance requirements of X.400 (1988) with respect  
to *protocol*.
- 69 009(D) If the product implements P1 (1988):  
70 The manufacturer and product shall satisfy the conformance  
71 requirements of X.400 (1988) with respect to P1 (1988). (See  
72 GA03 in 3.3.1.1.)
- 73 010(D) If the product implements P3 (1988):  
74 The manufacturer and product shall satisfy the conformance  
75 requirements of X.400 (1988) with respect to P3 (1988). (See  
76 GA03 in 3.3.1.1.)
- 77 011(D) If the product implements P7 (1988):  
78 The manufacturer and product shall satisfy the conformance  
79 requirements of X.400 (1988) with respect to P7 (1988). (See  
80 GA03 in 3.3.1.1.)
- 81 D09 If the product plays the role of service for the MA or MT interface:  
82 The manufacturer shall state the behavior of implementation-  
83 defined options.

### 84 3.3.1.2 Conformance Documentation

85 There are no test assertions for this subclause.

### 86 3.3.1.3 Conforming Implementation Options

- 87 D10 The circumstances that define abnormal termination of a session shall  
88 be documented.
- 89 D11 For *funct* of *ma\_open*, *mt\_open*:  
90 The features of a session that *funct* supplies as defaults shall be  
91 documented.
- 92 D12 Documentation shall state whether the service shall accept any cir-  
93 cumstances as causes of temporary delivery failure.
- 94 D13 The maximum number of users that may be assigned to a single  
95 delivery queue shall be documented.
- 96 D14 The maximum number of sessions that may exist simultaneously  
97 shall be documented.
- 98 D15 The documentation shall state which qualifying object in the delivery  
99 queue (if there are several such objects) *ma\_start\_delivery* selects for  
100 delivery.
- 101 D16 The documentation shall state which qualifying object in the input  
102 queue (if there are several such objects) *ma\_start\_transfer\_in* selects  
103 for transfer in.

104       D17       For *funct* of *ma\_wait*, *mt\_wait*:  
105                The documentation shall state to which and how many sessions  
106                *funct* notifies an arrival of an object when several sessions pro-  
107                vide access to a delivery, retrieval or input queue.

108       **3.3.2 Application Conformance**

109       There are no test assertions for this subclause.

110       **3.4 Test Assertions**

111       There are no test assertions for this clause.

1     **Section 4: Test Assertions for Terminology and General Requirements**

2     There are no test assertions for this section.

## 1                                    **Section 5: Test Assertions for Overview**

### 2                    **5.1 Purpose**

3                    There are no test assertions for this clause.

### 4                    **5.2 Overview**

- 5                    001(C)    If the MA interface is implemented:  
6                                    Three persistent databases—a submission queue, a delivery  
7                                    queue, and a retrieval queue—are maintained by the service  
8                                    even when it is out of contact with the client.
- 9                    002(C)    If the MT interface is implemented:  
10                                   Two persistent databases—an input queue, and an output  
11                                   queue—are maintained by the service even when it is out of  
12                                   contact with the client.

### 13                   **5.3 Object Management**

14                   There are no test assertions for this clause.

### 15                   **5.4 Features**

- 16                   003(A)    The interfaces enable the client and service to negotiate the use of the  
17                   functional units and packages of the MA and MT interfaces.
- 18                   004(A)    An implementation shall provide the following names to identify the  
19                   features of the MA and MT interfaces.

|    | Feature           | Name              |
|----|-------------------|-------------------|
| 20 | Basic Access FU   | basic-access      |
| 21 | Submission FU     | submission        |
| 22 | Delivery FU       | delivery          |
| 23 | Retrieval FU      | retrieval         |
| 24 | Basic Transfer FU | basic-transfer    |
| 25 | Transfer In FU    | transfer-in       |
| 26 | Transfer Out FU   | transfer-out      |
| 27 | IM 84 Package     | im-84             |
| 28 | IM 88 Package     | im-88             |
| 29 | MH 84 Package     | mh-84             |
| 30 | MH 88 Package     | mh-88             |
| 31 | SM 88 Package     | sm-88             |
| 32 | Multiple Delivery | multiple-delivery |
| 33 | General Content   | general-content   |

- 34        005(C)    If the Multiple Delivery feature is not supported:  
 35                When `ma_start_delivery` delivers a message or report, it shall  
 36                restrict itself to one recipient for each invocation.
- 37        GA04    For *feature* of Submission FU, Delivery FU, Retrieval FU:  
 38                When the client has not requested *feature* by means of a  
 39                parameter to the `ma_open` operation, then *feature* shall not be  
 40                available to the client.
- 41        GA05    For *feature* of Transfer In FU, Transfer Out FU:  
 42                When the client has not requested *feature* by means of a  
 43                parameter to the `mt_open` operation, then *feature* shall not be  
 44                available to the client.
- 45        GA06    For *feature* of Submission FU, Delivery FU, Retrieval FU:  
 46                If the Basic Access FU is supported and *feature* is supported:  
 47                When the Basic Access FU is available and *feature* is  
 48                unavailable and the client invokes a operation in *feature*,  
 49                then the `feature_unavailable` error arises.
- 50        006(A)    When the client supplies as an operation argument an object, one of  
 51                whose subobjects is an instance of a class that is not in an available  
 52                package, then the `no_such_class` error arises.
- 53        007(C)    If the Basic Access FU is supported and the Basic Transfer FU is  
 54                supported:  
 55                The Basic Access FU and the Basic Transfer FU are mutually  
 56                exclusive for a particular session between the client and the  
 57                service.
- 58        008(C)    If the Delivery FU is supported and the Retrieval FU is supported:  
 59                The Delivery FU and the Retrieval FU are mutually exclusive  
 60                for a particular session between the client and the service.

## 1        **Section 6: Test Assertions for Message Handling Interfaces**

### 2        **6.1 Datatypes**

#### 3        **6.1.1 Boolean List**

4            001(A)    When a data value of the datatype `mh_boolean_list_type` is defined,  
5                      then it is a list of Boolean values, used to indicate activated features.

#### 6        **6.1.2 Feature List**

7            002(A)    When a data value of the datatype `mh_feature_list_type` is defined,  
8                      then it is a list of `om_object_id_type` values, used to indicate  
9                      requested features.

#### 10       **6.1.3 Interval**

11           003(A)    When a data value of the datatype `mh_interval_type` is defined then, it  
12                      is an integer in the range  $0-2^{32}$ .

13           004(A)    When a data value of the datatype `mh_interval_type` is defined, then it  
14                      is an integer that denotes an interval of time measured in mil-  
15                      liseconds.

### 6.1.4 Object Count

- 16        005(A)    When a data value of the datatype `mh_object_count_type` is defined,  
17                then it is an integer in the range  $0-2^{32}$ .
- 18        006(A)    When a data value of the datatype `mh_object_count_type` is defined,  
19                then it is an integer that denotes a number of objects.
- 20        GA07      For all MA and MT supported *func* in `ma_size`, `mt_size`:  
21                When *func* is called and completes successfully, then it returns  
22                an integer of the datatype `mh_object_count_type`.

### 23        6.1.5 Sequence Number

- 24        007(A)    When a data value of the datatype `mh_sequence_number_type` is  
25                defined, then it is an integer in the range  $0-2^{32}$ .
- 26        008(A)    When a data value of the datatype `mh_sequence_number_type` is  
27                defined, then it is an integer that denotes the sequence number of an  
28                object in a retrieval queue.
- 29        009(A)    When a data value of the datatype `mh_sequence_number_type` is  
30                defined, then it is a unique integer assigned in ascending order.
- 31        010(A)    When `ma_start_retrieval` is called and completes successfully, then it  
32                returns an integer of the datatype `mh_sequence_number_type`.

## 33        6.2 Access Operations

- 34        R01        When `ma_cancel_submission` is called and completes successfully,  
35                then this operation cancels a submitted message, with the deferred  
36                delivery option.
- 37        R02        When `ma_close` is called and completes successfully, then this opera-  
38                tion terminates an MA session between the client and the service. If  
39                the delivery or retrieval of a message or a report is in progress, the  
40                service first unsuccessfully finishes that delivery or retrieval.
- 41        R03        When `ma_finish_delivery` is called and completes successfully, then  
42                this operation concludes the delivery in progress in a session. The  
43                client supplies delivery confirmations, as required, for users to which  
44                the object, a message, was delivered. It also indicates to which users  
45                the object, either a message or a report, is undeliverable.

- 46 R04 When `ma_finish_retrieval` is called and completes successfully, then  
 47 this operation concludes the retrieval in progress in a session. The  
 48 client indicates whether the service is to remove the retrieved mes-  
 49 sage, report from the retrieval queue, or leave it there.
- 49 R05 When `ma_open` is called and completes successfully, then this opera-  
 50 tion establishes an MA session between the client and the service, and  
 51 makes the Basic Access FU and the OM Package initially available in  
 52 that session. The client may also specify the other features required  
 53 for the session. The client specifies either its own name or the O/R  
 54 address of a local user. If the client's own name is specified, then the  
 55 session provides MTS access to a group of local users statically associ-  
 56 ated with the client name. If the O/R address of a local user is  
 57 specified, then the session provides MTS access to the local user at the  
 58 specified address.
- 59 R06 When `ma_size` is called and completes successfully, then this opera-  
 60 tion determines the number of unreserved objects in the input queue  
 61 to which a session provides access. Each object is a message or a  
 62 report.
- 63 R07 When `ma_start_delivery` is called and completes successfully, then this  
 64 operation begins the delivery of a message or a report to one or more  
 65 of the users associated with a session, by reserving an unreserved  
 66 object in the delivery queue. If no messages or reports await delivery,  
 67 the operation reports an exception.
- 68 R08 When `ma_start_retrieval` is called and completes successfully, then  
 69 this operation begins the retrieval of a message or a report, by reserv-  
 70 ing an unreserved object in the retrieval queue to which a session pro-  
 71 vides access. If no messages or reports await retrieval, the operation  
 72 reports an exception.
- 73 R09 When `ma_submit` is called and completes successfully, then this  
 74 operation submits a communicate, by adding it to the submission  
 75 queue to which a session provides access. The operation first verifies  
 76 the integrity of the communicate.
- 77 R10 When `ma_wait` is called and completes successfully, then this opera-  
 78 tion returns when a message or a report is available for delivery in  
 79 the delivery queue, retrieval in the retrieval queue to which a session  
 80 provides access, or when a period of time elapses, whichever occurs  
 81 first.
- 82 001(A) The MA operations are grouped into four FUs (one basic and one each  
 83 for submission, delivery, and retrieval) as indicated in the table  
 84 below:

| Basic Access | Submission        | Delivery        | Retrieval        |
|--------------|-------------------|-----------------|------------------|
| Open         | Submit            | Size            | Size             |
| Close        | Cancel Submission | Start Delivery  | Start Retrieval  |
| OM API       |                   | Finish Delivery | Finish Retrieval |
|              |                   | Wait            | Wait             |

## NOTES:

- 90 (1) The Delivery and Retrieval FUs are mutually exclusive.  
 91 (2) The OM API is defined in ISO/IEC 14360 {2}.
- 92 002(A) When the client has not requested Submission FU by means of a  
 93 parameter to the `ma_open` function, then Submission FU will not be  
 94 available to the client. (See GA04 in 5.4.)
- 95 003(C) If the Basic Access FU is supported and Submission FU is supported:  
 96 When the Basic Access FU is available and Submission FU is  
 97 unavailable and the client invokes a function in Submission  
 98 FU, then the `feature_unavailable` error arises. (See GA06 in  
 99 5.4.)
- 100 004(A) When the client has not requested Delivery FU by means of a param-  
 101 eter to the `ma_open` function, then Delivery FU will not be available  
 102 to the client. (See GA04 in 5.4.)
- 103 005(C) If the Basic Access FU is supported and Delivery FU is supported:  
 104 When the Basic Access FU is available and Delivery FU is un-  
 105 available and the client invokes an operation in Delivery FU,  
 106 then the `feature_unavailable` error arises. (See GA06 in 5.4.)
- 107 006(A) When the client has not requested Retrieval FU by means of a param-  
 108 eter to the `ma_open` function, then Retrieval FU will not be available  
 109 to the client. (See GA04 in 5.4.)
- 110 007(C) If the Basic Access FU is supported and Retrieval FU is supported:  
 111 When the Basic Access FU is available and Retrieval FU is  
 112 unavailable and the client invokes a function in Retrieval FU,  
 113 then the `feature_unavailable` error arises. (See GA06 in 5.4.)
- 114 008(C) If the product plays the role of service for Basic Access FU:  
 115 The product shall implement every aspect of every operation in  
 116 Basic Access FU. (See GA01 in 3.3.1.1.)
- 117 009(C) If the product plays the role of service for Submission FU:  
 118 The product shall implement every aspect of every operation in  
 119 Submission FU. (See GA01 in 3.3.1.1.)
- 120 010(C) If the product plays the role of service for Delivery FU:  
 121 The product shall implement every aspect of every operation in  
 122 Delivery FU. (See GA01 in 3.3.1.1.)
- 123 011(C) If the product plays the role of service for Retrieval FU:  
 124 The product shall implement every aspect of every operation in  
 125 Retrieval FU. (See GA01 in 3.3.1.1.)

## 6.2.1 Cancel Submission

### 126 6.2.1.1 Synopsis

127 There are no test assertions for this subclause.

### 128 6.2.1.2 Description

129 001(C) If the Submission FU is supported:  
130 When `ma_cancel_submission` is called and completes success-  
131 fully, then the deferred delivery of a submitted message is  
132 cancelled.

### 133 6.2.1.3 Results

134 002(C) If the Submission FU is supported:  
135 When `ma_cancel_submission` is called and does not complete  
136 successfully, then a return code indicating the reason is  
137 returned.

138 003(C) If the Submission FU is supported:  
139 When `ma_cancel_submission` is called and completes success-  
140 fully and no error condition applies, then it will return the  
141 success return code. (See GA26 in 6.4.)

### 142 6.2.1.4 Errors

143 004(C) If the Submission FU is supported:  
144 When `ma_cancel_submission` is called and the service does not  
145 offer the requested feature and no other error condition applies,  
146 then it will return either success or the `feature_unavailable`  
147 return code. (See GA10 in 6.4.)

148 005(D) If the Submission FU is supported:  
149 When `ma_cancel_submission` is called and the operation is  
150 aborted by an external force (e.g., a keystroke that is desig-  
151 nated for this purpose at a user interface) and no other error  
152 condition applies, then it will return success or the `function_-`  
153 `interrupted` return code. (See GA11 in 6.4.)

154 006(C) If the Submission FU is supported:  
155 When `ma_cancel_submission` is called and the service cannot  
156 allocate the main memory it needs to complete the operation  
157 and no other error condition applies, then it will return either

- 158 success or the `memory_insufficient` return code. (See GA12 in  
6.4.)
- 159 007(D) If the Submission FU is supported:  
160 When `ma_cancel_submission` is called and the service could not  
161 successfully employ the network upon which its implementa-  
162 tion depends and no other error condition applies, then it will  
163 return either success or the `network_error` return code. (See  
164 GA13 in 6.4.)
- 165 NOTE: This should be tested when the service is implemented over a net-  
166 work.
- 167 R01 If the Submission FU is supported:  
168 When `ma_cancel_submission(mts_identifier)` is called, a  
169 deferred delivery of a message cannot be canceled because the  
170 *mts\_identifier* does not identify a message previously submit-  
171 ted by the client, and no other error condition applies, then it  
172 will return either success or the `no_such_message` return  
173 code. [See 019(C) in 6.4.]
- 174 008(C) If the Submission FU is supported:  
175 When `ma_cancel_submission(session,)` is called and the  
176 specified *session* does not exist and no other error condition  
177 applies, then it will return either success or the `no_such_-`  
178 `session` return code. (See GA17 in 6.4.)
- 179 009(C) If the Submission FU is supported:  
180 When `ma_cancel_submission` is called and a purported syntax  
181 identifier is undefined and no other error condition applies,  
182 then it will return either success or the `no_such_syntax` return  
183 code. (See GA18 in 6.4.)
- 184 010(C) If the Submission FU is supported:  
185 When `ma_cancel_submission` is called and a purported data-  
186 type identifier is undefined and no other error condition  
187 applies, then it will return either success or the `no_such_type`  
188 return code. (See GA19 in 6.4.)
- 189 011(C) If the Submission FU is supported:  
190 When `ma_cancel_submission(object,)` is called and a public  
191 *object* has been specified and no other error condition applies,  
192 then it will return either success or the `not_private` return  
193 code. (See GA20 in 6.4.)
- 194 012(D) If the Submission FU is supported:  
195 When `ma_cancel_submission` is called and the service  
196 encounters a permanent difficulty other than those denoted by  
197 other return codes, then it will return either success or the  
198 `permanent_error` return code. (See GA22 in 6.4.)
- 199 013(D) If the Submission FU is supported:  
200 When `ma_cancel_submission` is called and the service could not  
201 successfully employ the operating system upon which its imple-  
202 mentation depends and no other error condition applies, then it

- will return either success or the `system_error` return code. (See GA27 in 6.4.)
- 203
- 204     014(D)   If the Submission FU is supported:
- 205             When `ma_cancel_submission` is called and the service
- 206             encounters a temporary difficulty other than those denoted by
- 207             other return codes, then it returns the `temporary_error` return
- 208             code. (See GA28 in 6.4.)
- 209     R02       If the Submission FU is supported:
- 210             When `ma_cancel_submission` is called and a deferred delivery
- 211             of a message has progressed for transfer or delivery, or the ser-
- 212             vice has provided the client with proof of submission and no
- 213             other error condition applies, then it will return either success
- 214             or the `too_late` return code. [See 023(C) in 6.4.]
- 215     015(D)   If the Submission FU is supported:
- 216             When `ma_cancel_submission` is called and an undefined imple-
- 217             mentation limit prevents the addition to an object of another
- 218             attribute value and no other error condition applies, then it will
- 219             return either success or the `too_many_values` return code. (See
- 220             GA30 in 6.4.)
- 221     016(C)   If the Submission FU is supported:
- 222             When `ma_cancel_submission(object)` is called and an *object* is
- 223             an instance of the wrong class and no other error condition
- 224             applies, then it will return either success or the `wrong_class`
- 225             return code. (See GA31 in 6.4.)
- 226     017(C)   If the Submission FU is supported:
- 227             When `ma_cancel_submission` is called and an attribute has, or
- 228             would have, a value that violates the value length constraints
- 229             in force and no other error condition applies, then it will return
- 230             either success or the `wrong_value_length` return code. (See
- 231             GA32 in 6.4.)
- 232     018(C)   If the Submission FU is supported:
- 233             When `ma_cancel_submission` is called and an attribute has, or
- 234             would have, a value that violates a constraint of the syntax of
- 235             the value and no other error condition applies, then it will
- 236             return either success or the `wrong_value_makeup` return code.
- 237             (See GA33 in 6.4.)
- 238     019(C)   If the Submission FU is supported:
- 239             When `ma_cancel_submission` is called and an attribute has, or
- 240             would have, a value that violates the value number constraints
- 241             in force and no other error condition applies, then it will return
- 242             either success or the `wrong_value_number` return code. (See
- 243             GA34 in 6.4.)
- 244     020(C)   If the Submission FU is supported:
- 245             When `ma_cancel_submission` is called and an attribute has, or
- 246             would have, a value whose syntax is not permitted and no
- 247             other error condition applies, then it will return either success
- 248             or the `wrong_value_syntax` return code. (See GA35 in 6.4.)

021(C) If the Submission FU is supported:

When `ma_cancel_submission` is called and an object has, or would have, a value whose type is not permitted and no other error condition applies, then it will return either `success` or the `wrong_value_type` return code. (See GA36 in 6.4.)

## 6.2.2 Close

### 6.2.2.1 Synopsis

There are no test assertions for this subclause.

### 6.2.2.2 Description

001(C) If the Basic Access FU is supported:

When `ma_close(session)` is called and completes successfully, then the specified MA *session* is terminated, as well as destroying the workspace.

002(C) If the Basic Access FU is supported:

When `ma_close(session)` is called and completes successfully, then any delivery or retrieval of a message or report occurring in the specified *session* has been unsuccessfully finished before the MA session is terminated and the workspace is destroyed.

### 6.2.2.3 Results

003(C) If the Basic Access FU is supported:

When `ma_close` is called and does not complete successfully, then a return code indicating the reason is returned.

004(C) If the Basic Access FU is supported:

When `ma_close` is called and completes successfully and no error condition applies, then it will return the `success` return code. (See GA26 in 6.4.)

#### 6.2.2.4 Errors

- 274        005(D)    If the Basic Access FU is supported:  
 275                When `ma_close` is called and the operation is aborted by an  
 276                external force (e.g., a keystroke designated for this purpose at a  
 277                user interface) and no other error condition applies, then it will  
 278                return success or the `function_interrupted` return code. (See  
 279                GA11 in 6.4.)
- 280        006(C)    If the Basic Access FU is supported:  
 281                When `ma_close` is called and the service cannot allocate the  
 282                main memory it needs to complete the operation and no other  
 283                error condition applies, then it will return either success or the  
 284                `memory_insufficient` return code. (See GA12 in 6.4.)
- 285        007(D)    If the Basic Access FU is supported:  
 286                When `ma_close` is called and the service could not successfully  
 287                employ the network upon which its implementation depends  
 288                and no other error condition applies, then it will return either  
 289                success or the `network_error` return code. (See GA13 in 6.4.)
- 290        008(C)    If the Basic Access FU is supported:  
 291                When `ma_close(session)` is called and the specified session does  
 292                not exist and no other error condition applies, then it will  
 293                return either success or the `no_such_session` return code. (See  
 294                GA17 in 6.4.)
- 295        009(C)    If the Basic Access FU is supported:  
 296                When `ma_close(session)` is called and a public object has been  
 297                specified and no other error condition applies, then it will  
 298                return either success or the `not_private` return code. (See GA20  
 299                in 6.4.)
- 300        010(D)    If the Basic Access FU is supported:  
 301                When `ma_close` is called and the service encounters a per-  
 302                manent difficulty other than those denoted by other return  
 303                codes, then it will return either success or the `permanent_error`  
 304                return code. (See GA22 in 6.4.)
- 305        011(D)    If the Basic Access FU is supported:  
 306                When `ma_close` is called and the service could not successfully  
 307                employ the operating system upon which its implementation  
 308                depends and no other error condition applies, then it will  
 309                return either success or the `system_error` return code. (See  
 310                GA27 in 6.4.)
- 311        012(D)    If the Basic Access FU is supported:  
 312                When `ma_close` is called and the service encounters a tem-  
 313                porary difficulty other than those denoted by other return  
 314                codes, then it returns the `temporary_error` return code. (See  
 315                GA28 in 6.4.)

013(C) If the Basic Access FU is supported:

316 When `ma_close(session)` is called and an object is the wrong  
 317 class and no other error condition applies, then it will return  
 318 either success or the `wrong_class` return code. (See GA31 in  
 319 6.4.)

## 320 6.2.3 Finish Delivery

### 321 6.2.3.1 Synopsis

322 There are no test assertions for this subclause.

### 323 6.2.3.2 Description

324 001(C) If the Delivery FU is supported:

325 When `ma_finish_delivery` is called and completes successfully,  
 326 then the delivery in progress in a session is concluded.

327 002(C) If the Delivery FU is supported:

328 When `ma_finish_delivery` is called and completes successfully  
 329 and no delivery or nondelivery confirmations have been  
 330 specified, then the service assumes that the message has been  
 331 delivered to the to the recipients not covered by a confirmation  
 332 or nondelivery.

333 D18 If the Delivery FU is supported:

334 When `ma_finish_delivery` is called, then whether the service  
 335 will accept any circumstances as causes of temporary failure is  
 336 service documented.

337 003(C) If the Delivery FU is supported:

338 When `ma_finish_delivery` is called and the service does not sup-  
 339 port temporary failures, then it treats them as permanent.

340 004(C) If the Delivery FU is supported:

341 When `ma_finish_delivery` is called and the object is a message,  
 342 then the service issues Delivery Reports for the users to which  
 343 it has been delivered.

344 005(C) If the Delivery FU is supported:

345 When `ma_finish_delivery` is called and the object is a message,  
 346 then the service issues Nondelivery Reports for the users to  
 347 which it is permanently undeliverable.

348 006(C) If the Delivery FU is supported:

349 When `ma_finish_delivery` is called and the object is a message,  
 350 then the service returns the message to the delivery queue for  
 351 the users to which it is temporarily undeliverable and marks  
 352 the object unreserved.

007(C) If the Delivery FU is supported:

353 When `ma_finish_delivery` is called and the object is a report and  
354 the user to which it was to be delivered is temporarily  
355 undelivered, then the service returns the report to the delivery  
356 queue.

008(C) If the Delivery FU is supported:

357 When `ma_finish_delivery` returns, the object handle remains  
358 valid.  
359

### 360 6.2.3.3 Arguments In

009(C) If the Delivery FU is supported:

361 When `ma_finish_delivery(,delivery_confirmations,)` is called,  
362 then *delivery\_confirmations* is one or more delivery  
363 confirmations, as required, for users to which the object, a mes-  
364 sage, was delivered.  
365

010(C) If the Delivery FU is supported:

366 When `ma_finish_delivery(,non_delivery_reports)` is called, then  
367 *non\_delivery\_reports* indicates one or more users to which the  
368 object, a message, cannot be delivered, or the single user to  
369 which the object, a report, cannot be delivered.  
370

### 371 6.2.3.4 Results

011(C) If the Delivery FU is supported:

372 When `ma_finish_delivery` is called and does not complete suc-  
373 cessfully, then a return code indicating the reason is returned.  
374

012(C) If the Delivery FU is supported:

375 When `ma_finish_delivery` is called and completes successfully  
376 and no error condition applies, then it will return the success  
377 return code. (See GA26 in 6.4.)  
378

### 379 6.2.3.5 Errors

013(C) If the Delivery FU is supported:

380 When `ma_finish_delivery` is called and the and the service does  
381 not offer the requested feature and no other error condition  
382 applies, then it will return either success or the feature\_-  
383 unavailable return code. (See GA10 in 6.4.)  
384

## 014(D) If the Delivery FU is supported:

When `ma_finish_delivery` is called and the operation is aborted by an external force (e.g., a keystroke at a user interface that is designated for this purpose) and no other error condition applies, then it will return success or the `function_interrupted` return code. (See GA11 in 6.4.)

## 015(C) If the Delivery FU is supported:

When `ma_finish_delivery` is called and the service cannot allocate the main memory it needs to complete the operation and no other error condition applies, then it will return either success or the `memory_insufficient` return code. (See GA12 in 6.4.)

## 016(D) If the Delivery FU is supported:

When `ma_finish_delivery` is called and the service could not successfully employ the network upon which its implementation depends and no other error condition applies, then it will return either success or the `network_error` return code. (See GA13 in 6.4.)

## 017(C) If the Delivery FU is supported:

When `ma_finish_delivery` is called and a purported object is nonexistent or the handle of the purported object is invalid and no other error condition applies, then it will return either success or the `no_such_object` return code. (See GA16 in 6.4.)

## 018(C) If the Delivery FU is supported:

When `ma_finish_delivery(session,,)` is called and the specified session does not exist and no other error condition applies, then it will return either success or the `no_such_session` return code. (See GA17 in 6.4.)

## 019(C) If the Delivery FU is supported:

When `ma_finish_delivery` is called and a purported syntax identifier is undefined and no other error condition applies, then it will return either success or the `no_such_syntax` return code. (See GA18 in 6.4.)

## 020(C) If the Delivery FU is supported:

When `ma_finish_delivery` is called and a purported type identifier is undefined and no other error condition applies, then it will return either success or the `no_such_type` return code. (See GA19 in 6.4.)

## 021(C) If the Delivery FU is supported:

When `ma_finish_delivery` is called and a public object has been specified and no other error condition applies, then it will return either success or the `not_private` return code. (See GA20 in 6.4.)

## 022(D) If the Delivery FU is supported:

When `ma_finish_delivery` is called and the service encounters a permanent difficulty other than those denoted by other return

- 430 codes, then it will return either `success` or the `permanent_error`  
return code. (See GA22 in 6.4.)
- 431 023(C) If the Delivery FU is supported:  
432 When `ma_finish_delivery` is called and a delivery or retrieval is  
433 not in progress in the session and no other error condition  
434 applies, then it will return either `success` or the `session_not_`  
435 `busy` return code. (See GA25 in 6.4.)
- 436 024(D) If the Delivery FU is supported:  
437 When `ma_finish_delivery` is called and the service could not suc-  
438 cessfully employ the operating system upon which its imple-  
439 mentation depends and no other error condition applies, then it  
440 will return either `success` or the `system_error` return code. (See  
441 GA27 in 6.4.)
- 442 025(D) If the Delivery FU is supported:  
443 When `ma_finish_delivery` is called and the service encounters a  
444 temporary difficulty other than those denoted by other return  
445 codes then it returns the `temporary_error` return code. (See  
446 GA28 in 6.4.)
- 447 026(D) If the Delivery FU is supported:  
448 When `ma_finish_delivery` is called and an undefined implemen-  
449 tation limit prevents the addition to an object of another attri-  
450 bute value and no other error condition applies, then it will  
451 return either `success` or the `too_many_values` return code. (See  
452 GA30 in 6.4.)
- 453 027(C) If the Delivery FU is supported:  
454 When `ma_finish_delivery` is called and an object is an instance  
455 of the wrong class and no other error condition applies, then it  
456 will return either `success` or the `wrong_class` return code. (See  
457 GA31 in 6.4.)
- 458 028(C) If the Delivery FU is supported:  
459 When `ma_finish_delivery` is called and an attribute has, or  
460 would have, a value that violates the value length constraints  
461 in force and no other error condition applies, then it will return  
462 either `success` or the `wrong_value_length` return code. (See  
463 GA32 in 6.4.)
- 464 029(C) If the Delivery FU is supported:  
465 When `ma_finish_delivery` is called and an attribute has, or  
466 would have, a value that violates a constraint of the syntax of  
467 the value, and no other error condition applies, then it will  
468 return either `success` or the `wrong_value_makeup` return code.  
469 (See GA33 in 6.4.)
- 470 030(C) If the Delivery FU is supported:  
471 When `ma_finish_delivery` is called and an attribute has, or  
472 would have, a value that violates the value number constraints  
473 in force and no other error condition applies, then it will return  
474 either `success` or the `wrong_value_number` return code. (See  
475 GA34 in 6.4.)

031(C) If the Delivery FU is supported:

476 When `ma_finish_delivery` is called and an attribute has, or  
 477 would have, a value whose syntax is not permitted and no  
 478 other error condition applies, then it will return either success  
 479 or the `wrong_value_syntax` return code. (See GA35 in 6.4.)

032(C) If the Delivery FU is supported:

480 When `ma_finish_delivery` is called and an object has, or would  
 481 have, a value whose type is not permitted and no other error  
 482 condition applies, then it will return either success or the  
 483 `wrong_value_type` return code (See GA36 in 6.4.)  
 484

## 6.2.4 Finish Retrieval

### 6.2.4.1 Synopsis

There are no test assertions for this subclause.

### 6.2.4.2 Description

001(C) If the Retrieval FU is supported:

489 When `ma_finish_retrieval` is called and completes successfully,  
 490 then it concludes the retrieval in progress in the specified ses-  
 491 sion.  
 492

002(C) If the Retrieval FU is supported:

493 When `ma_finish_retrieval(session,remove)` is called and `remove`  
 494 is specified to be true and the operation completes successfully,  
 495 then the service removes the message or report in progress  
 496 from the retrieval queue.  
 497

003(C) If the Retrieval FU is supported:

498 When `ma_finish_retrieval(session,remove)` is called and `remove`  
 499 is specified to be true, then the associated object handle  
 500 remains accessible.  
 501

004(C) If the Retrieval FU is supported:

502 When `ma_finish_retrieval(session,remove)` is called and `remove`  
 503 is specified to be false and completes successfully, then the mes-  
 504 sage or report in progress is left in the retrieval queue.  
 505

005(C) If the Retrieval FU is supported:

506 When `ma_finish_retrieval(session,remove)` is called and `remove`  
 507 is specified to be false and completes successfully, then the mes-  
 508 sage or report in progress is marked unreserved and made  
 509 inaccessible by the service.  
 510

### 6.2.4.3 Results

- 511        006(C)    If the Retrieval FU is supported:  
 512                When `ma_finish_retrieval` is called and does not complete suc-  
 513                cessfully, then a return code indicating the reason is returned.
- 514        007(C)    If the Retrieval FU is supported:  
 515                When `ma_finish_retrieval` is called and completes successfully  
 516                and no error condition applies, then it will return the success  
 517                return code. (See GA26 in 6.4.)

### 518        6.2.4.4 Errors

- 519        008(C)    If the Retrieval FU is supported:  
 520                When `ma_finish_retrieval` is called and the and the service does  
 521                not offer the requested feature and no other error condition  
 522                applies, then it will return either success or the `feature_-`  
 523                unavailable return code. (See GA10 in 6.4.)
- 524        009(D)    If the Retrieval FU is supported:  
 525                When `ma_finish_retrieval` is called and the operation is aborted  
 526                by an external force (e.g., a keystroke at user interface that is  
 527                designated for this purpose) and no other error condition  
 528                applies, then it will return success or the `function_interrupted`  
 529                return code. (See GA11 in 6.4.)
- 530        010(C)    If the Retrieval FU is supported:  
 531                When `ma_finish_retrieval` is called and the service cannot allo-  
 532                cate the main memory it needs to complete the operation and  
 533                no other error condition applies, then it will return either  
 534                success or the `memory_insufficient` return code. (See GA12 in  
 535                6.4.)
- 536        011(D)    If the Retrieval FU is supported:  
 537                When `ma_finish_retrieval` is called and the service could not suc-  
 538                cessfully employ the network upon which its implementation  
 539                depends and no other error condition applies, then it will  
 540                return either success or the `network_error` return code. (See  
 541                GA13 in 6.4.)
- 542        012(C)    If the Retrieval FU is supported:  
 543                When `ma_finish_retrieval` is called and a purported object is  
 544                nonexistent or the handle of the purported object is invalid and  
 545                no other error condition applies, then it will return either  
 546                success or the `no_such_object` return code. (See GA16 in 6.4.)
- 547        013(C)    If the Retrieval FU is supported:  
 548                When `ma_finish_retrieval(session,)` is called and the specified  
 549                session does not exist and no other error condition applies, then

it will return either success or the no\_such\_session return code. (See GA17 in 6.4.)

014(C) If the Retrieval FU is supported:

When ma\_finish\_retrieval is called and a public object has been specified and no other error condition applies, then it will return either success or the not\_private return code. (See GA20 in 6.4.)

015(D) If the Retrieval FU is supported:

When ma\_finish\_retrieval is called and the service encounters a permanent difficulty other than those denoted by other return codes, then it will return either success or the permanent\_error return code. (See GA22 in 6.4.)

016(C) If the Retrieval FU is supported:

When ma\_finish\_retrieval is called and a delivery or retrieval is not in progress in the session and no other error condition applies, then it will return either success or the session\_not\_busy return code. (See GA25 in 6.4.)

017(D) If the Retrieval FU is supported:

When ma\_finish\_retrieval is called and the service could not successfully employ the operating system upon which its implementation depends and no other error condition applies, then it will return either success or the system\_error return code. (See GA27 in 6.4.)

018(D) If the Retrieval FU is supported:

When ma\_finish\_retrieval is called and the service encounters a temporary difficulty other than those denoted by other return codes, then it returns the temporary\_error return code. (See GA28 in 6.4.)

019(C) If the Retrieval FU is supported:

When ma\_finish\_retrieval is called and an object is an instance of the wrong class and no other error condition applies, then it will return either success or the wrong\_class return code. (See GA31 in 6.4.)

## 6.2.5 Open

### 6.2.5.1 Synopsis

There are no test assertions for this subclause.

### 6.2.5.2 Description

- 585        001(C)    If the Basic Access FU is supported:  
 586                When `ma_open` is called and completes successfully, then an  
 587                MA session between the client and the service is established.
- 588        002(C)    If the Basic Access FU is supported:  
 589                When `ma_open` is called and completes successfully, then the  
 590                Basic Access FU and the OM Package is made initially avail-  
 591                able for the specified session.
- 592        D19        If the Basic Access FU is supported:  
 593                When `ma_open` is called, then the maximum number of users  
 594                in a group is documented.
- 595        003(C)    If the Basic Access FU is supported:  
 596                When `ma_open(,,,workspace)` is called and completes success-  
 597                fully, then the workspace *workspace* is created.
- 598        D20        If the Basic Access FU is supported:  
 599                When `ma_open` is called, then the maximum number of ses-  
 600                sions that may exist simultaneously is documented.
- 601        004(C)    If the Basic Access FU is supported:  
 602                When `ma_open` generates an instance of MH Class **Session**  
 603                and the subject domain is an ADMD, then `ma_open` will set  
 604                the **ADMD-Name** attribute to the name of the subject domain  
 605                relative to the country that the **Country-Name** attribute  
 606                denotes.
- 607                Testing Requirements: Test that the value of the **ADMD-**  
 608                **Name** attribute is taken from the administration data held by  
 609                the local MTA. (See GA399 in 8.2.42.)
- 610        005(C)    If the Basic Access FU is supported:  
 611                When `ma_open` generates an instance of MH Class **Session**  
 612                and the subject domain is a PRMD, then `ma_open` will set the  
 613                **ADMD-Name** attribute to the name of the ADMD to which the  
 614                subject domain is attached, relative to the country that the  
 615                **Country-Name** attribute denotes.
- 616                Testing Requirements: Test that the value of the **ADMD-**  
 617                **Name** attribute is taken from the administration data held by  
 618                the local MTA. (See GA400 in 8.2.42.)
- 619        006(C)    If the Basic Access FU is supported:  
 620                When `ma_open` generates an instance of MH Class **Session**,  
 621                then `ma_open` will set the **Country-Name** attribute to the  
 622                name of the country of the ADMD that the **ADMD-Name** attri-  
 623                bute denotes.

Testing Requirements: Test that the value of the **Country-Name** attribute is taken from the administration data held by the local MTA. (See GA401 in 8.2.42.)

007(C) If the Basic Access FU is supported:

When `ma_open` generates an instance of MH Class **Session** and sets the **Country-Name** attribute, then `ma_open` will set the **Country-Name** attribute to a value that is either the number that CCITT Recommendation X.121: 1984 {6} assigns to the country, or the character pair that ISO 3166: 1993 {1} assigns to it. (See GA402 in 8.2.42.)

008(C) If the Basic Access FU is supported:

When `ma_open` generates an instance of MH Class **Session** and the subject domain is a PRMD, then `ma_open` will set the **PRMD-Identifier** attribute to identify the PRMD relative to the ADMD that the **ADMD-Name** attribute denotes.

Testing Requirements: Test that the value of the **PRMD-Identifier** attribute is taken from the administration data held by the local MTA. (See GA403 in 8.2.42.)

### 6.2.5.3 Arguments In

### 6.2.5.4 Results

009(C) If the Basic Access FU is supported:

When `ma_open` is called and does not complete successfully, then a return code indicating the reason is returned.

010(C) If the Basic Access FU is supported:

When `ma_open` is called and completes successfully, and no error condition applies, then it will return the success return code. (See GA26 in 6.4.)

011(C) If the Basic Access FU is supported:

When `ma_open` is called and completes successfully, then *activated*, an ordered sequence of Boolean values with the same number of elements as the *feature\_list*, is returned.

012(C) If the Basic Access FU is supported:

When `ma_open` is called, then a true value in the *activated* result indicates that the corresponding feature is now a part of the interface.

013(C) If the Basic Access FU is supported:

When `ma_open` is called, then a **false** value in the *activated* result indicates that the corresponding feature is not available.

- 661       014(C)   If the Basic Access FU is supported:  
 662               When `ma_open(,,,*session,)` is called and completes success-  
 663               fully, then `*session` is the established MA session between the  
               client and service; it is an instance of the **session** class.
- 664       015(C)   If the Basic Access FU is supported:  
 665               When `ma_open(,,,*session,)` is called and completes success-  
 666               fully, then the service prevents `*session` from being modified by  
 667               the client.
- 668       016(D)   If the Basic Access FU is supported:  
 669               When `ma_open(,,,*session,)` is called and does not complete  
 670               successfully, then `*session` is not present.
- 671       017(C)   If the Basic Access FU is supported:  
 672               When `ma_open(,,,*workspace)` is called and completes success-  
 673               fully, then `*workspace`, which will contain all objects returned  
 674               as a result of the operations invoked in the session, is returned.

#### 675       6.2.5.5 Errors

- 676       018(C)   If the Basic Access FU is supported:  
 677               When `ma_open(,feature_list,,)` is called and the requested  
 678               feature conflicts with an existing feature of the session and no  
 679               other error condition applies, then it will return either success  
 680               or the `feature_conflicts` return code. (See GA09 in 6.4.)
- 681       019(C)   If the Basic Access FU is supported:  
 682               When `ma_open` is called and the and the service does not offer  
 683               the requested feature and no other error condition applies, then  
 684               it will return either success or the `feature_unavailable` return  
 685               code. (See GA10 in 6.4.)
- 686       020(D)   If the Basic Access FU is supported:  
 687               When `ma_open` is called and the operation is aborted by an  
 688               external force (e.g., a keystroke at a user interface that is desig-  
 689               nated for this purpose) and no other error condition applies,  
 690               then it will return success or the `function_interrupted` return  
 691               code. (See GA11 in 6.4.)
- 692       021(C)   If the Basic Access FU is supported:  
 693               When `ma_open` is called and the service cannot allocate the  
 694               main memory it needs to complete the operation and no other  
 695               error condition applies, then it will return either success or the  
 696               `memory_insufficient` return code. (See GA12 in 6.4.)
- 697       022(D)   If the Basic Access FU is supported:  
 698               When `ma_open` is called and the service could not successfully  
 699               employ the network upon which its implementation depends

- and no other error condition applies, then it will return either success or the network\_error return code. (See GA13 in 6.4.)
- 700
- 701       023(C)   If the Basic Access FU is supported:
- 702               When `ma_open(client_name,,)` is called and the service does
- 703               not recognize the specified client name and no other error con-
- 704               dition applies, then it will return either success or the no\_-
- 705               such\_client return code. (See GA15 in 6.4.)
- 706       024(C)   If the Basic Access FU is supported:
- 707               When `ma_open` is called and a purported syntax identifier is
- 708               undefined and no other error condition applies, then it will
- 709               return either success or the no\_such\_syntax return code. (See
- 710               GA18 in 6.4.)
- 711       025(C)   If the Basic Access FU is supported:
- 712               When `ma_open` is called and a purported type identifier is
- 713               undefined and no other error condition applies, then it will
- 714               return either success or the no\_such\_type return code. (See
- 715               GA19 in 6.4.)
- 716       R01       If the Basic Access FU is supported:
- 717               When `ma_open(user_address,,)` is called and no other error
- 718               condition applies, then it will return either success or the no\_-
- 719               such\_user return code. [See 020(C) in 6.4.]
- 720       026(D)   If the Basic Access FU is supported:
- 721               When `ma_open` is called and the service encounters a per-
- 722               manent difficulty other than those denoted by other return
- 723               codes, then it will return either success or the permanent\_error
- 724               return code. (See GA22 in 6.4.)
- 725       027(D)   If the Basic Access FU is supported:
- 726               When `ma_open` is called and the service could not successfully
- 727               employ the operating system upon which its implementation
- 728               depends and no other error condition applies, then it will
- 729               return either success or the system\_error return code. (See
- 730               GA27 in 6.4.)
- 731       028(D)   If the Basic Access FU is supported:
- 732               When `ma_open` is called and the service encounters a tem-
- 733               porary difficulty other than those denoted by other return
- 734               codes, then it returns the temporary\_error return code. (See
- 735               GA28 in 6.4.)
- 736       029(C)   If the Basic Access FU is supported:
- 737               When `ma_open` is called and the number of sessions is equal to
- 738               an implementation defined limit and no other error condition
- 739               applies, then it will return either success or the too\_many\_-
- 740               sessions return code. (See GA29 in 6.4.)
- 741       030(D)   If the Basic Access FU is supported:
- 742               When `ma_open` is called and an undefined implementation
- 743               limit prevents the addition to an object of another attribute
- 744               value and no other error condition applies, then it will return

either success or the `too_many_values` return code. (See GA30 in 6.4.)

745

746 001(C) If the Basic Access FU is supported:

747 When `ma_open` is called and an object is an instance of the  
748 wrong class and no other error condition applies, then it will  
749 return either success or the `wrong_class` return code. (See  
750 GA31 in 6.4.)

751 002(C) If the Basic Access FU is supported:

752 When `ma_open` is called and an attribute has, or would have, a  
753 value that violates the value length constraints in force and no  
754 other error condition applies, then it will return either success  
755 or the `wrong_value_length` return code. (See GA32 in 6.4.)

756 003(C) If the Basic Access FU is supported:

757 When `ma_open` is called and an attribute has, or would have, a  
758 value that violates a constraint of the syntax of the value and  
759 no other error condition applies, then it will return either  
760 success or the `wrong_value_makeup` return code. (See GA33 in  
761 6.4.)

762 004(C) If the Basic Access FU is supported:

763 When `ma_open` is called and an attribute has, or would have,  
764 a value that violates the value number constraints in force and  
765 no other error condition applies, then it will return either  
766 success or the `wrong_value_number` return code. (See GA34 in  
767 6.4.)

768 005(C) If the Basic Access FU is supported:

769 When `ma_open` is called and an attribute has, or would have, a  
770 value whose syntax is not permitted and no other error condi-  
771 tion applies, then it will return either success or the `wrong_-`  
772 `value_syntax` return code. (See GA35 in 6.4.)

773 006(C) If the Basic Access FU is supported:

774 When `ma_open` is called and an object has, or would have, a  
775 value whose type is not permitted and no other error condition  
776 applies, then it will return either success or the `wrong_value_-`  
777 `type` return code. (See GA36 in 6.4.)

## 778 6.2.6 Size

### 779 6.2.6.1 Synopsis

780 There are no test assertions for this subclause.

### 781 6.2.6.2 Description

782 001(C) If the Delivery FU is supported:

783 When `ma_size(*number)` is called and completes successfully,

then it returns the number of messages and reports in the delivery queue of the specified session.

002(C) If the Retrieval FU is supported:

When *ma\_size* is called and completes successfully, then it returns the number of messages and reports in the retrieval queue of the specified session.

### 6.2.6.3 Results

003(C) If the Delivery or Retrieval FU is supported:

When *ma\_size* is called and completes successfully and no error condition applies, then it will return the success return code. (See GA26 in 6.4.)

004(C) If the Delivery or Retrieval FU is supported:

When *ma\_size* is called and does not complete successfully, then a return code indicating the reason is returned.

005(C) If the Delivery FU is supported:

When *ma\_size* is called and completes successfully, then *number* indicating the number of unreserved objects in the delivery queue will be returned.

006(C) If the Retrieval FU is supported:

When *ma\_size* is called and completes successfully, then *number* indicating the number of unreserved objects in the retrieval queue will be returned.

007(C) If the Delivery or Retrieval FU is supported:

When *ma\_size* is called and completes successfully and *number* returned exceeds  $2^{16} - 1$ , then  $2^{16} - 1$  is returned.

008(D) If the Delivery or Retrieval FU is supported:

When *ma\_size* is called and does not complete successfully, then *number* will not be present.

009(C) If the Delivery or Retrieval FU is supported:

When *ma\_size* is called and completes successfully, then it returns an integer of the type *mh\_object\_count\_type*. (See GA07 in 6.1.4.)

### 6.2.6.4 Errors

010(C) If the Delivery or Retrieval FU is supported:

When *ma\_size* is called and the service does not offer the requested feature and no other error condition applies, then

- 819 it will return either success or the `feature_unavailable` return  
code. (See GA10 in 6.4.)
- 820 011(D) If the Delivery or Retrieval FU is supported:  
821 When `ma_size` is called and the operation is aborted by an  
822 external force (e.g., a keystroke at a user interface that is desig-  
823 nated for this purpose) and no other error condition  
824 applies, then it will return success or the `function_interrupted`  
825 return code. (See GA11 in 6.4.)
- 826 012(C) If the Delivery or Retrieval FU is supported:  
827 When `ma_size` is called and the service cannot allocate the  
828 main memory it needs to complete the operation and no other  
829 error condition applies, then it will return either success or the  
830 `memory_insufficient` return code. (See GA12 in 6.4.)
- 831 013(D) If the Delivery or Retrieval FU is supported:  
832 When `ma_size` is called and the service could not successfully  
833 employ the network upon which its implementation depends  
834 and no other error condition applies, then it will return either  
835 success or the `network_error` return code. (See GA13 in 6.4.)
- 836 014(C) If the Delivery or Retrieval FU is supported:  
837 When `ma_size(session, )` is called and the specified session does  
838 not exist and no other error condition applies, then it will  
839 return either success or the `no_such_session` return code. (See  
840 GA17 in 6.4.)
- 841 015(C) If the Delivery or Retrieval FU is supported:  
842 When `ma_size` is called and a public object has been specified  
843 and no other error condition applies, then it will return either  
844 success or the `not_private` return code. (See GA20 in 6.4.)
- 845 016(D) If the Delivery or Retrieval FU is supported:  
846 When `ma_size` is called and the service encounters a per-  
847 manent difficulty other than those denoted by other return  
848 codes then it will return either success or the `permanent_error`  
849 return code. (See GA22 in 6.4.)
- 850 017(D) If the Delivery or Retrieval FU is supported:  
851 When `ma_size` is called and the service could not successfully  
852 employ the operating system upon which its implementation  
853 depends and no other error condition applies, then it will  
854 return either success or the `system_error` return code. (See  
855 GA27 in 6.4.)
- 856 018(D) If the Delivery or Retrieval FU is supported:  
857 When `ma_size` is called and the service encounters a temporary  
858 difficulty other than those denoted by other return codes, then  
859 it returns the `temporary_error` return code. (See GA28 in 6.4.)
- 860 019(C) If the Delivery or Retrieval FU is supported:  
861 When `ma_size` is called and an object is an instance of the  
862 wrong class and no other error condition applies, then it will

return either success or the wrong\_class return code. (See GA31 in 6.4.)

## 6.2.7 Start Delivery

### 6.2.7.1 Synopsis

There are no test assertions for this subclause.

### 6.2.7.2 Description

001(C) If the Delivery FU is supported:

When ma\_start\_delivery is called, then it begins the delivery of a message or a report to one or more of the users associated with a session.

002(C) If the Delivery FU is supported:

When ma\_start\_delivery is called and completes successfully, then an unreserved object in the delivery queue is reserved.

003(C) If the Delivery FU is supported:

When ma\_start\_delivery is called and no objects await delivery, then an exception is returned.

004(C) If the Delivery FU is supported:

When ma\_start\_delivery is called, then the delivery of an object cannot simultaneously be in progress in any other session.

005(C) If the Delivery FU is supported:

If the multiple-delivery feature for a session is supported:

When ma\_start\_delivery is called and an object is addressed to several users, then the service begins delivery in one operation invocation.

Otherwise:

When ma\_start\_delivery is called and an object is addressed to several users, then the service begins delivery in one operation invocation per user.

D21 If the Delivery FU is supported:

When ma\_start\_delivery is called and there are several objects in the delivery queue, then the object qualifying for selection by the service is documented.

### 6.2.7.3 Results

006(C) If the Delivery FU is supported:

When ma\_start\_delivery is called and does not complete

- 897 successfully, then a return code indicating the reason is  
returned.
- 898 007(C) If the Delivery FU is supported:  
899 When `ma_start_delivery` is called and completes successfully  
900 and no error condition applies, then it will return the success  
901 return code. (See GA26 in 6.4.)
- 902 008(C) If the Delivery FU is supported:  
903 When `ma_start_delivery(*object)` is called and completes suc-  
904 cessfully, then *\*object* is an instance of the **Delivered-**  
905 **Message** class with an envelope for each one or more of the  
906 users associated with the session whose delivery has started.
- 907 009(C) If the Delivery FU is supported:  
908 When `ma_start_delivery(*object)` is called and completes suc-  
909 cessfully, then *\*object* is an instance of the **Delivered-Report**  
910 class whose delivery has started.
- 911 010(C) If the Delivery FU is supported:  
912 When `ma_start_delivery(*object)` is called and completes suc-  
913 cessfully, then *\*object* is present.
- 914 011(D) If the Delivery FU is supported:  
915 When `ma_start_delivery(*object)` is called and does not com-  
916 plete successfully, then *\*object* is not present.
- 917 012(D) If the IM 88 Package is supported or the IM 84 Package is supported  
918 and the Delivery FU is supported:  
919 When `ma_start_delivery` generates an instance of IM Class  
920 **Bilaterally-Defined-Body-Part**, then it will set the  
921 **Bilateral-Data** attribute to the binary data that the body part  
922 is intended to convey in the message.
- 923 Testing Requirements: The value of the **Bilateral-Data** attri-  
924 bute will be determined from the conveyed message. (See  
925 GA37 in 7.3.1.)
- 926 013(D) If the IM 88 Package is supported and the Delivery FU is supported:  
927 When `ma_start_delivery` generates an instance of IM Class  
928 **Externally-Defined-Body-Part**, then it will set the  
929 **External-Data** attribute to the information object that the  
930 body part is intended to convey in the message.
- 931 Testing Requirements: The value of the **External-Data** attri-  
932 bute will be determined from the conveyed message. (See  
933 GA39 in 7.3.3.)
- 934 014(D) If the IM 88 Package is supported and the Delivery FU is supported:  
935 When `ma_start_delivery` generates an instance of IM Class  
936 **Externally-Defined-Body-Part** and the body part contains  
937 external parameters, then `ma_start_delivery` will set the

- 938                   **External-Parameters** attribute to the information object that  
characterizes the value of the External Data attribute.
- 939                   Testing Requirements: The value of the **External-**  
940 **Parameters** attribute will be determined from the conveyed  
941 message. (See GA41 in 7.3.3.)
- 942           015(D)   If the IM 88 Package is supported or the IM 84 Package is supported  
943                   and the Delivery FU is supported:  
944                   When `ma_start_delivery` generates an instance of IM Class  
945 **G3-Fax-Body-Part** and the body part contains the NBPs of  
946 the images, then `ma_start_delivery` will set the **G3-Fax-NBPs**  
947 attribute to the NBPs of the images.
- 948                   Testing Requirements: The value of the **G3-Fax-NBPs** attri-  
949 bute will be determined from the conveyed message. (See  
950 GA43 in 7.3.4.)
- 951           016(D)   If the IM 88 Package is supported or the IM 84 Package is supported  
952                   and the Delivery FU is supported:  
953                   When `ma_start_delivery` generates an instance of IM Class  
954 **G3-Fax-Body-Part**, then `ma_start_delivery` will set the  
955 **Images** attribute to the G3 facsimile image(s) in the body part.
- 956                   Testing Requirements: The values of the **Images** attribute will  
957 be determined from the conveyed message. (See GA45 in  
958 7.3.4.)
- 959           017(D)   If the IM 88 Package is supported or the IM 84 Package is supported  
960                   and the Delivery FU is supported:  
961                   When `ma_start_delivery` generates an instance of IM Class  
962 **G4-Class-1-Body-Part**, then `ma_start_delivery` will set the  
963 **G4-Class-1-Document** attribute to the final form document(s)  
964 of the body part.
- 965                   Testing Requirements: The values of the **G4-Class-1-**  
966 **Document** attribute will be determined from the conveyed  
967 message. (See GA47 in 7.3.5.)
- 968           018(D)   If the IM 88 Package is supported and the Delivery FU is supported:  
969                   When `ma_start_delivery` generates an instance of IM Class  
970 **General-Text-Body-Part**, then `ma_start_delivery` will set the  
971 **Char-Set-Registration** attribute to the list of the character  
972 set registrations that are or may be present in the data com-  
973 ponent of the body part.

- 974                   Testing Requirements: The values of the **Char-Set-Registration** attribute will be determined from the conveyed  
975                   message. (See GA49 in 7.3.6.)
- 976           019(D)   If the IM 88 Package is supported and the Delivery FU is supported:  
977                   When `ma_start_delivery` generates an instance of IM Class  
978                   **General-Text-Body-Part**, then `ma_start_delivery` will set the  
979                   **General-Text-Data** attribute to all of the text of the body  
980                   part.
- 981                   Testing Requirements: The value of the **General-Text-Data**  
982                   attribute will be determined from the conveyed message. (See  
983                   GA51 in 7.3.6.)
- 984           020(D)   If the IM 88 Package is supported or the IM 84 Package is supported  
985                   and the Delivery FU is supported:  
986                   When `ma_start_delivery` generates an instance of IM Class  
987                   **IA5-Text-Body-Part**, then `ma_start_delivery` will set the  
988                   **Repertoire** attribute to identify the character set to which the  
989                   text of the body part is constrained.
- 990                   Testing Requirements: The value of the **Repertoire** attribute  
991                   will be determined from the conveyed message. (See GA53 in  
992                   7.3.7.)
- 993           021(D)   If the IM 88 Package is supported or the IM 84 Package is supported  
994                   and the Delivery FU is supported:  
995                   When `ma_start_delivery` generates an instance of IM Class  
996                   **IA5-Text-Body-Part**, then `ma_start_delivery` will set the **Text**  
997                   attribute to all of the text of the body part.
- 998                   Testing Requirements: The value of the **Text** attribute will be  
999                   determined from the conveyed message. (See GA55 in 7.3.7.)
- 1000          022(D)   If the IM 88 Package is supported or the IM 84 Package is supported  
1001                   and the Delivery FU is supported:  
1002                   When `ma_start_delivery` generates an instance of IM Class  
1003                   **Interpersonal-Message** and the authorizing users are other  
1004                   than the originator of the IPM alone, then `ma_start_delivery`  
1005                   will set the **Authorizing-Users** attribute to the O/R descrip-  
1006                   tors for the authorizing users of the interpersonal message.
- 1007                   Testing Requirements: The values of the **Authorizing-Users**  
1008                   attribute will be determined from the conveyed message. (See  
1009                   GA57 in 7.3.8.)

- 1010 023(C) If the IM 88 Package is supported or the IM 84 Package is supported  
 1011 and the Delivery FU is supported:  
 1012 When `ma_start_delivery` generates an instance of IM Class  
 1013 **Interpersonal-Message** and the authorizing user is the IPM's  
 1014 originator alone, then `ma_start_delivery` will not set the  
**Authorizing-Users** attribute. (See GA58 in 7.3.8.)
- 1015 024(D) If the IM 88 Package is supported or the IM 84 Package is supported  
 1016 and the Delivery FU is supported:  
 1017 When `ma_start_delivery` generates an instance of IM Class  
 1018 **Interpersonal-Message**, then `ma_start_delivery` will set the  
 1019 **Auto-forwarded** attribute to indicate whether the IPM is the  
 1020 result of automatic (not manual) forwarding.
- 1021 Testing Requirements: The values of the **Auto-forwarded**  
 1022 attribute will be determined from the conveyed message. (See  
 1023 GA60 in 7.3.8.)
- 1024 025(D) If the IM 88 Package is supported and the Delivery FU is supported:  
 1025 When `ma_start_delivery` generates an instance of IM Class  
 1026 **Interpersonal-Message**, then `ma_start_delivery` will set the  
 1027 **Auto-submitted** attribute to indicate whether the IPM was  
 1028 submitted without human intervention and, if so, whether the  
 1029 message was auto-generated or auto-replied.
- 1030 Testing Requirements: The values of the **Auto-submitted**  
 1031 attribute will be determined from the conveyed message.  
 1032 (See GA62 in 7.3.8.)
- 1033 026(D) If the IM 88 Package is supported or the IM 84 Package is supported  
 1034 and the Delivery FU is supported:  
 1035 When `ma_start_delivery` generates an instance of IM Class  
 1036 **Interpersonal-Message** and the subject IPM instance is not  
 1037 intended for a primary or copy recipient, then `ma_start_delivery`  
 1038 will set the **Blind-Copy-Recipients** attribute to the recipient  
 1039 specifiers for blind copy recipients of the inter-personal mes-  
 1040 sage.
- 1041 Testing Requirements: The values of the **Blind-Copy-**  
 1042 **Recipients** attribute will be determined from the conveyed  
 1043 message. (See GA64 in 7.3.8.)
- 1044 027(C) If the IM 88 Package is supported or the IM 84 Package is supported  
 1045 and the Delivery FU is supported:  
 1046 When `ma_start_delivery` generates an instance of IM Class  
 1047 **Interpersonal-Message** and the subject IPM instance is  
 1048 intended for a primary or copy recipient, then `ma_start_delivery`  
 1049 will not set the **Blind-Copy-Recipients** attribute. (See GA65  
 1050 in 7.3.8.)

- 1051 028(D) If the IM 88 Package is supported or the IM 84 Package is supported  
 1052 and the Delivery FU is supported:  
 1053 When `ma_start_delivery` generates an instance of IM Class  
 1054 **Interpersonal-Message**, then `ma_start_delivery` will set the  
 1055 **Body** attribute to the information objects that the IPM is  
 1056 intended to convey from its originator to its intended recipients,  
 one information object per attribute value.
- 1057 Testing Requirements: The values of the **Body** attribute will  
 1058 be determined from the conveyed message. (See GA67 in  
 1059 7.3.8.)
- 1060 029(D) If the IM 88 Package is supported or the IM 84 Package is supported  
 1061 and the Delivery FU is supported:  
 1062 When `ma_start_delivery` generates an instance of IM Class  
 1063 **Interpersonal-Message** and there are copy recipients for the  
 1064 IPM, then `ma_start_delivery` will set the **Copy-Recipients**  
 1065 attribute to the recipient specifiers for the copy recipients of the  
 1066 IPM.
- 1067 Testing Requirements: The values of the **Copy-Recipients**  
 1068 attribute will be determined from the conveyed message. (See  
 1069 GA69 in 7.3.8.)
- 1070 030(C) If the IM 88 Package is supported or the IM 84 Package is supported  
 1071 and the Delivery FU is supported:  
 1072 When `ma_start_delivery` generates an instance of IM Class  
 1073 **Interpersonal-Message** and there are no copy recipients for  
 1074 the IPM, then `ma_start_delivery` will not set the **Copy-**  
 1075 **Recipients** attribute. (See GA70 in 7.3.8.)
- 1076 031(D) If the IM 88 Package is supported or the IM 84 Package is supported  
 1077 and the Delivery FU is supported:  
 1078 When `ma_start_delivery` generates an instance of IM Class  
 1079 **Interpersonal-Message** and the originator of the IPM opted  
 1080 for an expiry time to be set, then `ma_start_delivery` will set the  
 1081 **Expiry-Time** attribute to the date and time at which the  
 1082 authorizing users consider the IPM to lose its validity.
- 1083 Testing Requirements: The value of the **Expiry-Time** attri-  
 1084 bute will be determined from the conveyed message. (See  
 1085 GA72 in 7.3.8.)
- 1086 032(D) If the IM 88 Package is supported or the IM 84 Package is supported  
 1087 and the Delivery FU is supported:  
 1088 When `ma_start_delivery` generates an instance of IM Class  
 1089 **Interpersonal-Message**, then `ma_start_delivery` will set the

- 1090                    **Importance** attribute to identify the importance that the  
authorizing users attach to the IPM.
- 1091                    Testing Requirements: The value of the **Importance** attribute  
1092                    will be determined from the conveyed message. (See GA74 in  
1093                    7.3.8.)
- 1094            033(D)   If the IM 88 Package is supported and the Delivery FU is supported:  
1095                    When `ma_start_delivery` generates an instance of IM Class  
1096                    **Interpersonal-Message**, then `ma_start_delivery` will set the  
1097                    **Incomplete-Copy** attribute to indicate whether one or more  
1098                    values of the **Body** attribute, or all values of another attribute,  
1099                    are absent from the present instance of the IPM.
- 1100                    Testing Requirements: The value of the **Incomplete-Copy**  
1101                    attribute will be determined from the conveyed message. (See  
1102                    GA76 in 7.3.8.)
- 1103            034(D)   If the IM 88 Package is supported and the Delivery FU is supported:  
1104                    When `ma_start_delivery` generates an instance of IM Class  
1105                    **Interpersonal-Message** and the originator has included any  
1106                    IPM extended heading fields in the IPM, then `ma_start_delivery`  
1107                    will set the **IPM-Extensions** attribute to hold these exten-  
1108                    sions.
- 1109                    Testing Requirements: The values of the **IPM-Extensions**  
1110                    attribute will be determined from the conveyed message.  
1111                    (See GA78 in 7.3.8.)
- 1112            035(D)   If the IM 88 Package is supported and the Delivery FU is supported:  
1113                    When `ma_start_delivery` generates an instance of IM Class  
1114                    **Interpersonal-Message** and the originator has opted to iden-  
1115                    tify the languages used in the composition of the **Body** and  
1116                    **Subject** attributes, then `ma_start_delivery` will set the  
1117                    **Languages** attribute to identify these languages.
- 1118                    Testing Requirements: The values of the **Languages** attribute  
1119                    will be determined from the conveyed message. (See GA80 in  
1120                    7.3.8.)
- 1121            036(D)   If the IM 88 Package is supported or the IM 84 Package is supported  
1122                    and the Delivery FU is supported:  
1123                    When `ma_start_delivery` generates an instance of IM Class  
1124                    **Interpersonal-Message** and the originator of the IPM opted  
1125                    to indicate IPMs obsoleted by the present IPM, then `ma_start_`  
1126                    `delivery` will set the **Obsoleted-IPMs** attribute to the IPM