

Network Working Group  
Request for Comments: 1327  
Obsoletes: RFCs 987, 1026, 1138, 1148  
Updates: RFC 822

S. Hardcastle-Kille  
University College London  
May 1992

## Mapping between X.400(1988) / ISO 10021 and RFC 822

### Status of this Memo

This RFC specifies an IAB standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "IAB Official Protocol Standards" for the standardization state and status of this protocol. Distribution of this memo is unlimited.

### Abstract

This document describes a set of mappings which will enable interworking between systems operating the CCITT X.400 1988) Recommendations on Message Handling Systems / ISO IEC 10021 Message Oriented Text Interchange Systems (MOTIS) [CCITT/ISO88a], and systems using the RFC 822 mail protocol [Crocker82a] or protocols derived from RFC 822. The approach aims to maximise the services offered across the boundary, whilst not requiring unduly complex mappings. The mappings should not require any changes to end systems. This document is a revision based on RFCs 987, 1026, 1138, and 1148 [Kille86a,Kille87a] which it obsoletes.

This document specifies a mapping between two protocols. This specification should be used when this mapping is performed on the DARPA Internet or in the UK Academic Community. This specification may be modified in the light of implementation experience, but no substantial changes are expected.

### Table of Contents

1	- Overview .....	3
1.1	- X.400 .....	3
1.2	- RFC 822 .....	3
1.3	- The need for conversion .....	4
1.4	- General approach .....	4
1.5	- Gatewaying Model .....	5
1.6	- X.400 (1984) .....	8
1.7	- Compatibility with previous versions .....	8
1.8	- Aspects not covered .....	8
1.9	- Subsetting .....	9
1.10	- Document Structure .....	9

1.11	- Acknowledgements .....	9
2	- Service Elements .....	10
2.1	- The Notion of Service Across a Gateway .....	10
2.2	- RFC 822 .....	11
2.3	- X.400 .....	15
3	- Basic Mappings .....	24
3.1	- Notation .....	24
3.2	- ASCII and IA5 .....	26
3.3	- Standard Types .....	26
3.4	- Encoding ASCII in Printable String .....	28
4	- Addressing .....	30
4.1	- A textual representation of MTS.ORAddress .....	30
4.2	- Basic Representation .....	31
4.3	- EBNF.822-address <-> MTS.ORAddress .....	36
4.4	- Repeated Mappings .....	48
4.5	- Directory Names .....	50
4.6	- MTS Mappings .....	50
4.7	- IPMS Mappings .....	55
5	- Detailed Mappings .....	59
5.1	- RFC 822 -> X.400 .....	59
5.2	- Return of Contents .....	67
5.3	- X.400 -> RFC 822 .....	67
Appendix A	- Mappings Specific to SMTP .....	91
Appendix B	- Mappings specific to the JNT Mail .....	91
1	- Introduction .....	91
2	- Domain Ordering .....	91
3	- Addressing .....	91
4	- Acknowledge-To: .....	91
5	- Trace .....	92
6	- Timezone specification .....	92
7	- Lack of 822-MTS originator specification .....	92
Appendix C	- Mappings specific to UUCP Mail .....	93
Appendix D	- Object Identifier Assignment .....	94
Appendix E	- BNF Summary .....	94
Appendix F	- Format of address mapping tables .....	101
1	- Global Mapping Information .....	101
2	- Syntax Definitions .....	102
3	- Table Lookups .....	103
4	- Domain -> O/R Address format .....	104
5	- O/R Address -> Domain format .....	104
6	- Domain -> O/R Address of Gateway table .....	104
Appendix G	- Mapping with X.400(1984) .....	105
Appendix H	- RFC 822 Extensions for X.400 access .....	106
Appendix I	- Conformance .....	106
Appendix J	- Change History: RFC 987, 1026, 1138, 1148 .....	107
1	- Introduction .....	108
2	- Service Elements .....	108
3	- Basic Mappings .....	108

4	- Addressing .....	108
5	- Detailed Mappings .....	109
6	- Appendices .....	109
Appendix K	- Change History: RFC 1148 to this Document .....	109
1	- General .....	109
2	- Basic Mappings .....	110
3	- Addressing .....	110
4	- Detailed Mappings .....	110
5	- Appendices .....	110
References	.....	111
Security Considerations	.....	113
Author's Address	.....	113

## Chapter 1 -- Overview

### 1.1. X.400

This document relates to the CCITT 1988 X.400 Series Recommendations / ISO IEC 10021 on the Message Oriented Text Interchange Service (MOTIS). This ISO/CCITT standard is referred to in this document as "X.400", which is a convenient shorthand. Any reference to the 1984 CCITT Recommendations will be explicit. X.400 defines an Interpersonal Messaging System (IPMS), making use of a store and forward Message Transfer System. This document relates to the IPMS, and not to wider application of X.400. It is expected that X.400 will be implemented very widely.

### 1.2. RFC 822

RFC 822 evolved as a messaging standard on the DARPA (the US Defense Advanced Research Projects Agency) Internet. It specifies and end to end message format. It is used in conjunction with a number of different message transfer protocol environments.

#### SMTP Networks

On the DARPA Internet and other TCP/IP networks, RFC 822 is used in conjunction with two other standards: RFC 821, also known as Simple Mail Transfer Protocol (SMTP) [Postel82a], and RFC 920 which is a Specification for domains and a distributed name service [Postel84a].

#### UUCP Networks

UUCP is the UNIX to UNIX CoPy protocol, which is usually used over dialup telephone networks to provide a simple message transfer mechanism. There are some extensions to RFC 822, particularly in the addressing. They use domains which conform to RFC 920, but not the corresponding domain nameservers [Horton86a].

#### Bitnet

Some parts of Bitnet and related networks use RFC 822 related protocols, with EBCDIC encoding.

#### JNT Mail Networks

A number of X.25 networks, particularly those associated with the UK Academic Community, use the JNT (Joint Network Team) Mail Protocol, also known as Greybook [Kille84a]. This is used with domains and name service specified by the JNT NRS (Name Registration Scheme) [Larmouth83a].

The mappings specified here are appropriate for all of these networks.

### 1.3. The need for conversion

There is a large community using RFC 822 based protocols for mail services, who will wish to communicate with users of the IPMS provided by X.400 systems. This will also be a requirement in cases where communities intend to make a transition to use of an X.400 IPMS, as conversion will be needed to ensure a smooth service transition. It is expected that there will be more than one gateway, and this specification will enable them to behave in a consistent manner. Note that the term gateway is used to describe a component performing the protocol mappings between RFC 822 and X.400. This is standard usage amongst mail implementors, but should be noted carefully by transport and network service implementors.

Consistency between gateways is desirable to provide:

1. Consistent service to users.
2. The best service in cases where a message passes through multiple gateways.

### 1.4. General approach

There are a number of basic principles underlying the details of the specification. These principles are goals, and are not achieved in all aspects of the specification.

1. The specification should be pragmatic. There should not be a requirement for complex mappings for "Academic" reasons. Complex mappings should not be required to support trivial additional functionality.
2. Subject to 1), functionality across a gateway should be as high as possible.

3. It is always a bad idea to lose information as a result of any transformation. Hence, it is a bad idea for a gateway to discard information in the objects it processes. This includes requested services which cannot be fully mapped.
4. All mail gateways actually operate at exactly one level above the layer on which they conceptually operate. This implies that the gateway must not only be cognisant of the semantics of objects at the gateway level, but also be cognisant of higher level semantics. If meaningful transformation of the objects that the gateway operates on is to occur, then the gateway needs to understand more than the objects themselves.
5. Subject to 1), the specification should be reversible. That is, a double transformation should bring you back to where you started.

### 1.5. Gatewaying Model

#### 1.5.1. X.400

X.400 defines the IPMS Abstract Service in X.420/ISO 10021-7, [CCITT/ISO88b] which comprises of three basic services:

1. Origination
2. Reception
3. Management

Management is a local interaction between the user and the IPMS, and is therefore not relevant to gatewaying. The first two services consist of operations to originate and receive the following two objects:

1. IPM (Interpersonal Message). This has two components: a heading, and a body. The body is structured as a sequence of body parts, which may be basic components (e.g., IA5 text, or G3 fax), or IP Messages. The heading consists of fields containing end to end user information, such as subject, primary recipients (To:), and importance.
2. IPN (Inter Personal Notification). A notification about receipt of a given IPM at the UA level.

The Origination service also allows for origination of a probe, which is an object to test whether a given IPM could be correctly received.

The Reception service also allows for receipt of Delivery Reports (DR), which indicate delivery success or failure.

These IPMS Services utilise the Message Transfer (MT) Abstract Service [CCITT/ISO88c]. The MT Abstract Service provides the following three basic services:

1. Submission (used by IPMS Origination)
2. Delivery (used by IPMS Reception)
3. Administration (used by IPMS Management)

Administration is a local issue, and so does not affect this standard. Submission and delivery relate primarily to the MTS Message (comprising Envelope and Content), which carries an IPM or IPN (or other uninterpreted contents). There is also an Envelope, which includes an ID, an originator, and a list of recipients. Submission also includes the probe service, which supports the IPMS Probe. Delivery also includes Reports, which indicate whether a given MTS Message has been delivered or not.

The MTS is REFINED into the MTA (Message Transfer Agent) Service, which defines the interaction between MTAs, along with the procedures for distributed operation. This service provides for transfer of MTS Messages, Probes, and Reports.

#### 1.5.2. RFC 822

RFC 822 is based on the assumption that there is an underlying service, which is here called the 822-MTS service. The 822-MTS service provides three basic functions:

1. Identification of a list of recipients.
2. Identification of an error return address.
3. Transfer of an RFC 822 message.

It is possible to achieve 2) within the RFC 822 header. Some 822-MTS protocols, in particular SMTP, can provide additional functionality, but as these are neither mandatory in SMTP, nor available in other 822-MTS protocols, they are not considered here. Details of aspects specific to two 822-MTS protocols are given in Appendices B and C. An RFC 822 message consists of a header, and content which is uninterpreted ASCII text. The header is divided into fields, which are the protocol elements. Most of these fields are analogous to P2 heading fields, although some are analogous to MTS Service Elements

or MTA Service Elements.

### 1.5.3. The Gateway

Given this functional description of the two services, the functional nature of a gateway can now be considered. It would be elegant to consider the 822-MTS service mapping onto the MTS Service Elements and RFC 822 mapping onto an IPM, but reality just does not fit. Another elegant approach would be to treat this document as the definition of an X.400 Access Unit (AU). Again, reality does not fit. It is necessary to consider that the IPM format definition, the IPMS Service Elements, the MTS Service Elements, and MTA Service Elements on one side are mapped into RFC 822 + 822-MTS on the other in a slightly tangled manner. The details of the tangle will be made clear in Chapter 5. Access to the MTA Service Elements is minimised.

The following basic mappings are thus defined. When going from RFC 822 to X.400, an RFC 822 message and the associated 822-MTS information is always mapped into an IPM (MTA, MTS, and IPMS Services). Going from X.400 to RFC 822, an RFC 822 message and the associated 822-MTS information may be derived from:

1. A Report (MTA, and MTS Services)
2. An IPN (MTA, MTS, and IPMS services)
3. An IPM (MTA, MTS, and IPMS services)

Probes (MTA Service) must be processed by the gateway, as discussed in Chapter 5. MTS Messages containing Content Types other than those defined by the IPMS are not mapped by the gateway, and should be rejected at the gateway.

### 1.5.4. Repeated Mappings

The primary goal of this specification is to support single mappings, so that X.400 and RFC 822 users can communicate with maximum functionality.

The mappings specified here are designed to work where a message traverses multiple times between X.400 and RFC 822. This is often essential, particularly in the case of distribution lists. However, in general, this will lead to a level of service which is the lowest common denominator (approximately the services offered by RFC 822).

Some RFC 822 networks may wish to use X.400 as an interconnection mechanism (typically for policy reasons), and this is fully supported.

Where an X.400 messages transfers to RFC 822 and then back to X.400, there is no expectation of X.400 services which do not have an equivalent service in standard RFC 822 being preserved - although this may be possible in some cases.

#### 1.6. X.400 (1984)

Much of this work is based on the initial specification of RFC 987 and in its addendum RFC 1026, which defined a mapping between X.400(1984) and RFC 822. A basic decision is that the mapping defined in this document is to the full 1988 version of X.400, and not to a 1984 compatible subset. New features of X.400(1988) can be used to provide a much cleaner mapping than that defined in RFC 987. This is important, to give good support to communities which will utilise full X.400 at an early date. To interwork with 1984 systems, Appendix G shall be followed.

If a message is being transferred to an X.400(1984) system by way of X.400(1988) MTA it will give a slightly better service to follow the rules of Appendix G.

#### 1.7. Compatibility with previous versions

The changes between this and older versions of the document are given in Appendices I and J. These are RFCs 987, 1026, 1138, and 1148. This document is a revision of RFC 1148 [Kille90a]. As far as possible, changes have been made in a compatible fashion.

#### 1.8. Aspects not covered

There have been a number of cases where RFC 987 was used in a manner which was not intended. This section is to make clear some limitations of scope. In particular, this specification does not specify:

- Extensions of RFC 822 to provide access to all X.400 services
- X.400 user interface definition
- Mapping X.400 to extended versions of RFC 822, with support for multimedia content.

The first two of these are really coupled. To map the X.400 services, this specification defines a number of extensions to RFC 822. As a side effect, these give the 822 user access to SOME X.400 services. However, the aim on the RFC 822 side is to preserve current service, and it is intentional that access is not given to



all X.400 services. Thus, it will be a poor choice for X.400 implementors to use RFC 987(88) as an interface - there are too many aspects of X.400 which cannot be accessed through it. If a text interface is desired, a specification targeted at X.400, without RFC 822 restrictions, would be more appropriate. Some optional and limited extensions in this area have proved useful, and are defined in Appendix H.

### 1.9. Subsetting

This proposal specifies a mapping which is appropriate to preserve services in existing RFC 822 communities. Implementations and specifications which subset this specification are strongly discouraged.

### 1.10. Document Structure

This document has five chapters:

1. Overview - this chapter.
2. Service Elements - This describes the (end user) services mapped by a gateway.
3. Basic mappings - This describes some basic notation used in Chapters 3-5, the mappings between character sets, and some fundamental protocol elements.
4. Addressing - This considers the mapping between X.400 O/R names and RFC 822 addresses, which is a fundamental gateway component.
5. Detailed Mappings - This describes the details of all other mappings.

There are also eleven appendices.

#### WARNING:

THE REMAINDER OF THIS SPECIFICATION IS TECHNICALLY DETAILED. IT WILL NOT MAKE SENSE, EXCEPT IN THE CONTEXT OF RFC 822 AND X.400 (1988). DO NOT ATTEMPT TO READ THIS DOCUMENT UNLESS YOU ARE FAMILIAR WITH THESE SPECIFICATIONS.

### 1.11. Acknowledgements

The work in this specification was substantially based on RFC 987 and RFC 1148, which had input from many people, who are credited in the respective documents.

A number of comments from people on RFC 1148 lead to this document. In particular, there were comments and suggestions from: Maurice Abraham (HP); Harald Alvestrand (Sintef); Peter Cowen (X-Tel); Jim Craigie (JNT); Ella Gardener (MITRE); Christian Huitema (Inria); Erik Huizer (SURFnet); Neil Jones DEC); Ignacio Martinez (IRIS); Julian Onions (X-Tel); Simon Poole (SWITCH); Clive Roberts (Data General); Pete Vanderbilt SUN); Alan Young (Concurrent).

## Chapter 2 - Service Elements

This chapter considers the services offered across a gateway built according to this specification. It gives a view of the functionality provided by such a gateway for communication with users in the opposite domain. This chapter considers service mappings in the context of SINGLE transfers only, and not repeated mappings through multiple gateways.

### 2.1. The Notion of Service Across a Gateway

RFC 822 and X.400 provide a number of services to the end user. This chapter describes the extent to which each service can be supported across an X.400 <-> RFC 822 gateway. The cases considered are single transfers across such a gateway, although the problems of multiple crossings are noted where appropriate.

#### 2.1.1. Origination of Messages

When a user originates a message, a number of services are available. Some of these imply actions (e.g., delivery to a recipient), and some are insertion of known data (e.g., specification of a subject field). This chapter describes, for each offered service, to what extent it is supported for a recipient accessed through a gateway. There are three levels of support:

##### Supported

The corresponding protocol elements map well, and so the service can be fully provided.

##### Not Supported

The service cannot be provided, as there is a complete mismatch.

##### Partial Support

The service can be partially fulfilled.

In the first two cases, the service is simply marked as "Supported" or "Not Supported". Some explanation may be given if there are additional implications, or the (non) support is not intuitive. For

partial support, the level of partial support is summarised. Where partial support is good, this will be described by a phrase such as "Supported by use of.....". A common case of this is where the service is mapped onto a non-standard service on the other side of the gateway, and this would have lead to support if it had been a standard service. In many cases, this is equivalent to support. For partial support, an indication of the mechanism is given, in order to give a feel for the level of support provided. Note that this is not a replacement for Chapter 5, where the mapping is fully specified.

If a service is described as supported, this implies:

- Semantic correspondence.
- No (significant) loss of information.
- Any actions required by the service element.

An example of a service gaining full support: If an RFC 822 originator specifies a Subject: field, this is considered to be supported, as an X.400 recipient will get a subject indication.

In many cases, the required action will simply be to make the information available to the end user. In other cases, actions may imply generating a delivery report.

All RFC 822 services are supported or partially supported for origination. The implications of non-supported X.400 services is described under X.400.

#### 2.1.2. Reception of Messages

For reception, the list of service elements required to support this mapping is specified. This is really an indication of what a recipient might expect to see in a message which has been remotely originated.

#### 2.2. RFC 822

RFC 822 does not explicitly define service elements, as distinct from protocol elements. However, all of the RFC 822 header fields, with the exception of trace, can be regarded as corresponding to implicit RFC 822 service elements.

##### 2.2.1. Origination in RFC 822

A mechanism of mapping, used in several cases, is to map the RFC 822 header into a heading extension in the IPM (InterPersonal Message).

This can be regarded as partial support, as it makes the information available to any X.400 implementations which are interested in these services. Communities which require significant RFC 822 interworking are recommended to require that their X.400 User Agents are able to display these heading extensions. Support for the various service elements (headers) is now listed.

Date:

Supported.

From:

Supported. For messages where there is also a sender field, the mapping is to "Authorising Users Indication", which has subtly different semantics to the general RFC 822 usage of From:.

Sender:

Supported.

Reply-To:

Supported.

To: Supported.

Cc: Supported.

Bcc: Supported.

Message-Id:

Supported.

In-Reply-To:

Supported, for a single reference. Where multiple references are given, partial support is given by mapping to "Cross Referencing Indication". This gives similar semantics.

References:

Supported.

Keywords:

Supported by use of a heading extension.

Subject:

Supported.

Comments:

Supported by use of an extra body part.

**Encrypted:**

Supported by use of a heading extension.

**Resent-\***

Supported by use of a heading extension. Note that addresses in these fields are mapped onto text, and so are not accessible to the X.400 user as addresses. In principle, fuller support would be possible by mapping onto a forwarded IP Message, but this is not suggested.

**Other Fields**

In particular X-\* fields, and "illegal" fields in common usage (e.g., "Fruit-of-the-day:") are supported by use of heading extensions.

**2.2.2. Reception by RFC 822**

This considers reception by an RFC 822 User Agent of a message originated in an X.400 system and transferred across a gateway. The following standard services (headers) may be present in such a message:

Date:

From:

Sender:

Reply-To:

To:

Cc:

Bcc:

Message-Id:

In-Reply-To:

References:

Subject:

The following non-standard services (headers) may be present. These are defined in more detail in Chapter 5 (5.3.4, 5.3.6, 5.3.7):

Autoforwarded:  
Content-Identifier:  
Conversion:  
Conversion-With-Loss:  
Delivery-Date:  
Discarded-X400-IPMS-Extensions:  
Discarded-X400-MTS-Extensions:  
DL-Expansion-History:  
Deferred-Delivery:  
Expiry-Date:  
Importance:  
Incomplete-Copy:  
Language:  
Latest-Delivery-Time:  
Message-Type:  
Obsoletes:  
Original-Encoded-Information-Types:  
Originator-Return-Address:  
Priority:  
Reply-By:  
Requested-Delivery-Method:  
Sensitivity:  
X400-Content-Type:  
X400-MTS-Identifier:

X400-Originator:

X400-Received:

X400-Recipients:

### 2.3. X.400

#### 2.3.1. Origination in X.400

When mapping services from X.400 to RFC 822 which are not supported by RFC 822, new RFC 822 headers are defined. It is intended that these fields will be registered, and that co-operating RFC 822 systems may use them. Where these new fields are used, and no system action is implied, the service can be regarded as being partially supported. Chapter 5 describes how to map X.400 services onto these new headers. Other elements are provided, in part, by the gateway as they cannot be provided by RFC 822.

Some service elements are marked N/A (not applicable). There are five cases, which are marked with different comments:

##### N/A (local)

These elements are only applicable to User Agent / Message Transfer Agent interaction and so they cannot apply to RFC 822 recipients.

##### N/A (PDAU)

These service elements are only applicable where the recipient is reached by use of a Physical Delivery Access Unit (PDAU), and so do not need to be mapped by the gateway.

##### N/A (reception)

These services are only applicable for reception.

##### N/A (prior)

If requested, this service must be performed prior to the gateway.

##### N/A (MS)

These services are only applicable to Message Store (i.e., a local service).

Finally, some service elements are not supported. In particular, the new security services are not mapped onto RFC 822. Unless otherwise indicated, the behaviour of service elements marked as not supported will depend on the criticality marking supplied by the user. If the element is marked as critical for transfer or delivery, a non-

delivery notification will be generated. Otherwise, the service request will be ignored.

#### 2.3.1.1. Basic Interpersonal Messaging Service

These are the mandatory IPM services as listed in Section 19.8 of X.400 / ISO/IEC 10021-1, listed here in the order given. Section 19.8 has cross references to short definitions of each service.

Access management

N/A (local).

Content Type Indication

Supported by a new RFC 822 header (Content-Type:).

Converted Indication

Supported by a new RFC 822 header (X400-Received:).

Delivery Time Stamp Indication

N/A (reception).

IP Message Identification

Supported.

Message Identification

Supported, by use of a new RFC 822 header (X400-MTS-Identifier). This new header is required, as X.400 has two message-ids whereas RFC 822 has only one (see previous service).

Non-delivery Notification

Not supported, although in general an RFC 822 system will return error reports by use of IP messages. In other service elements, this pragmatic result can be treated as effective support of this service element.

Original Encoded Information Types Indication

Supported as a new RFC 822 header (Original-Encoded-Information-Types:).

Submission Time Stamp Indication

Supported.

Typed Body

Some types supported. IA5 is fully supported. ForwardedIPMessage is supported, with some loss of information. Other types get some measure of support, dependent on X.400 facilities for conversion to IA5. This



will only be done where content conversion is not prohibited.

User Capabilities Registration  
N/A (local).

#### 2.3.1.2. IPM Service Optional User Facilities

This section describes support for the optional (user selectable) IPM services as listed in Section 19.9 of X.400 / ISO/IEC 10021- 1, listed here in the order given. Section 19.9 has cross references to short definitions of each service.

Additional Physical Rendition  
N/A (PDAU).

Alternate Recipient Allowed  
Not supported. There is no RFC 822 service equivalent to prohibition of alternate recipient assignment (e.g., an RFC 822 system may freely send an undeliverable message to a local postmaster). Thus, the gateway cannot prevent assignment of alternative recipients on the RFC 822 side. This service really means giving the user control as to whether or not an alternate recipient is allowed. This specification requires transfer of messages to RFC 822 irrespective of this service request, and so this service is not supported.

Authorising User's Indication  
Supported.

Auto-forwarded Indication  
Supported as new RFC 822 header (Auto-Forwarded:).

Basic Physical Rendition  
N/A (PDAU).

Blind Copy Recipient Indication  
Supported.

Body Part Encryption Indication  
Supported by use of a new RFC 822 header (Original-Encoded-Information-Types:), although in most cases it will not be possible to map the body part in question.

Content Confidentiality  
Not supported.

Content Integrity  
Not supported.

Conversion Prohibition  
Supported. In this case, only messages with IA5 body parts, other body parts which contain only IA5, and Forwarded IP Messages (subject recursively to the same restrictions), will be mapped.

Conversion Prohibition in Case of Loss of Information  
Supported.

Counter Collection  
N/A (PDAU).

Counter Collection with Advice  
N/A (PDAU).

Cross Referencing Indication  
Supported.

Deferred Delivery  
N/A (prior). This service should always be provided by the MTS prior to the gateway. A new RFC 822 header Deferred-Delivery:) is provided to transfer information on this service to the recipient.

Deferred Delivery Cancellation  
N/A (local).

Delivery Notification  
Supported. This is performed at the gateway. Thus, a notification is sent by the gateway to the originator. If the 822-MTS protocol is JNT Mail, a notification may also be sent by the recipient UA.

Delivery via Bureaufax Service  
N/A (PDAU).

Designation of Recipient by Directory Name  
N/A (local).

Disclosure of Other Recipients  
Supported by use of a new RFC 822 header (X400-Recipients:). This is descriptive information for the RFC 822 recipient, and is not reverse mappable.

**DL Expansion History Indication**

Supported by use of a new RFC 822 header  
DL-Expansion-History:).

**DL Expansion Prohibited**

Distribution List means MTS supported distribution list, in the manner of X.400. This service does not exist in the RFC 822 world. RFC 822 distribution lists should be regarded as an informal redistribution mechanism, beyond the scope of this control. Messages will be sent to RFC 822, irrespective of whether this service is requested. Theoretically therefore, this service is supported, although in practice it may appear that it is not supported.

**Express Mail Service**

N/A (PDAU).

**Expiry Date Indication**

Supported as new RFC 822 header (Expiry-Date:). In general, no automatic action can be expected.

**Explicit Conversion**

N/A (prior).

**Forwarded IP Message Indication**

Supported, with some loss of information. The message is forwarded in an RFC 822 body, and so can only be interpreted visually.

**Grade of Delivery Selection**

N/A (PDAU)

**Importance Indication**

Supported as new RFC 822 header (Importance:).

**Incomplete Copy Indication**

Supported as new RFC 822 header (Incomplete-Copy:).

**Language Indication**

Supported as new RFC 822 header (Language:).

**Latest Delivery Designation**

Not supported. A new RFC 822 header (Latest-Delivery-Time:) is provided, which may be used by the recipient.

**Message Flow Confidentiality**

Not supported.

Message Origin Authentication

N/A (reception).

Message Security Labelling

Not supported.

Message Sequence Integrity

Not supported.

Multi-Destination Delivery

Supported.

Multi-part Body

Supported, with some loss of information, in that the structuring cannot be formalised in RFC 822.

Non Receipt Notification Request

Not supported.

Non Repudiation of Delivery

Not supported.

Non Repudiation of Origin

N/A (reception).

Non Repudiation of Submission

N/A (local).

Obsoleting Indication

Supported as new RFC 822 header (Obsoletes:).

Ordinary Mail

N/A (PDAU).

Originator Indication

Supported.

Originator Requested Alternate Recipient

Not supported, but is placed as comment next to address X400-Recipients:).

Physical Delivery Notification by MHS

N/A (PDAU).

Physical Delivery Notification by PDS

N/A (PDAU).

**Physical Forwarding Allowed**

Supported by use of a comment in a new RFC 822 header X400-Recipients:), associated with the recipient in question.

**Physical Forwarding Prohibited**

Supported by use of a comment in a new RFC 822 header X400-Recipients:), associated with the recipient in question.

**Prevention of Non-delivery notification**

Supported, as delivery notifications cannot be generated by RFC 822. In practice, errors will be returned as IP Messages, and so this service may appear not to be supported (see Non-delivery Notification).

**Primary and Copy Recipients Indication**

Supported

**Probe**

Supported at the gateway (i.e., the gateway services the probe).

**Probe Origin Authentication**

N/A (reception).

**Proof of Delivery**

Not supported.

**Proof of Submission**

N/A (local).

**Receipt Notification Request Indication**

Not supported.

**Redirection Allowed by Originator**

Redirection means MTS supported redirection, in the manner of X.400. This service does not exist in the RFC 822 world. RFC 822 redirection (e.g., aliasing) should be regarded as an informal redirection mechanism, beyond the scope of this control. Messages will be sent to RFC 822, irrespective of whether this service is requested. Theoretically therefore, this service is supported, although in practice it may appear that it is not supported.

**Registered Mail**

N/A (PDAU).

Registered Mail to Addressee in Person  
N/A (PDAU).

Reply Request Indication  
Supported as comment next to address.

Replying IP Message Indication  
Supported.

Report Origin Authentication  
N/A (reception).

Request for Forwarding Address  
N/A (PDAU).

Requested Delivery Method  
N/A (local). The services required must be dealt with at submission time. Any such request is made available through the gateway by use of a comment associated with the recipient in question.

Return of Content  
In principle, this is N/A, as non-delivery notifications are not supported. In practice, most RFC 822 systems will return part or all of the content along with the IP Message indicating an error (see Non-delivery Notification).

Sensitivity Indication  
Supported as new RFC 822 header (Sensitivity:).

Special Delivery  
N/A (PDAU).

Stored Message Deletion  
N/A (MS).

Stored Message Fetching  
N/A (MS).

Stored Message Listing  
N/A (MS).

Stored Message Summary  
N/A (MS).

Subject Indication  
Supported.

Undeliverable Mail with Return of Physical Message  
N/A (PDAU).

#### Use of Distribution List

In principle this applies only to X.400 supported distribution lists (see DL Expansion Prohibited). Theoretically, this service is N/A (prior). In practice, because of informal RFC 822 lists, this service can be regarded as supported.

### 2.3.2. Reception by X.400

#### 2.3.2.1. Standard Mandatory Services

The following standard IPM mandatory user facilities are required for reception of RFC 822 originated mail by an X.400 UA.

Content Type Indication

Delivery Time Stamp Indication

IP Message Identification

Message Identification

Non-delivery Notification

Original Encoded Information Types Indication

Submission Time Stamp Indication

Typed Body

#### 2.3.2.2. Standard Optional Services

The following standard IPM optional user facilities are required for reception of RFC 822 originated mail by an X.400 UA.

Authorising User's Indication

Blind Copy Recipient Indication

Cross Referencing Indication

Originator Indication

Primary and Copy Recipients Indication

Replying IP Message Indication

Subject Indication

#### 2.3.2.3. New Services

A new service "RFC 822 Header Field" is defined using the extension facilities. This allows for any RFC 822 header field to be represented. It may be present in RFC 822 originated messages, which are received by an X.400 UA.

### Chapter 3 Basic Mappings

#### 3.1. Notation

The X.400 protocols are encoded in a structured manner according to ASN.1, whereas RFC 822 is text encoded. To define a detailed mapping, it is necessary to refer to detailed protocol elements in each format. A notation to achieve this is described in this section.

##### 3.1.1. RFC 822

Structured text is defined according to the Extended Backus Naur Form (EBNF) defined in Section 2 of RFC 822 [Crocker82a]. In the EBNF definitions used in this specification, the syntax rules given in Appendix D of RFC 822 are assumed. When these EBNF tokens are referred to outside an EBNF definition, they are identified by the string "822." appended to the beginning of the string (e.g., 822.addr-spec). Additional syntax rules, to be used throughout this specification, are defined in this chapter.

The EBNF is used in two ways.

1. To describe components of RFC 822 messages (or of 822-MTS components). In this case, the lexical analysis defined in Section 3 of RFC 822 shall be used. When these new EBNF tokens are referred to outside an EBNF definition, they are identified by the string "EBNF." appended to the beginning of the string (e.g., EBNF.importance).
2. To describe the structure of IA5 or ASCII information not in an RFC 822 message. In these cases, tokens will either be self delimiting, or be delimited by self delimiting tokens. Comments and LWSP are not used as delimiters, except for the following cases, where LWSP may be inserted according to RFC 822 rules.



- Around the ":" in all headers
- EBNF.labelled-integer
- EBNF.object-identifier
- EBNF.encoded-info

RFC 822 folding rules are applied to all headers.

### 3.1.2. ASN.1

An element is referred to with the following syntax, defined in EBNF:

```

element      = service "." definition *( "." definition )
service      = "IPMS" / "MTS" / "MTA"
definition   = identifier / context
identifier    = ALPHA *< ALPHA or DIGIT or "-" >
context      = "[" 1*DIGIT "]"

```

The EBNF.service keys are shorthand for the following service specifications:

IPMS IPMSInformationObjects defined in Annex E of X.420 / ISO 10021-7.

MTS MTSAbstractService defined in Section 9 of X.411 / ISO 10021-4.

MTA MTAAbstractService defined in Section 13 of X.411 / ISO 10021-4.

The first EBNF.identifier identifies a type or value key in the context of the defined service specification. Subsequent EBNF.identifiers identify a value label or type in the context of the first identifier (SET or SEQUENCE). EBNF.context indicates a context tag, and is used where there is no label or type to uniquely identify a component. The special EBNF.identifier keyword "value" is used to denote an element of a sequence.

For example, IPMS.Heading.subject defines the subject element of the IPMS heading. The same syntax is also used to refer to element values. For example,

MTS.EncodedInformationTypes.[0].g3Fax refers to a value of MTS.EncodedInformationTypes.[0] .

### 3.2. ASCII and IA5

A gateway will interpret all IA5 as ASCII. Thus, mapping between these forms is conceptual.

### 3.3. Standard Types

There is a need to convert between ASCII text, and some of the types defined in ASN.1 [CCITT/ISO88d]. For each case, an EBNF syntax definition is given, for use in all of this specification, which leads to a mapping between ASN.1, and an EBNF construct. All EBNF syntax definitions of ASN.1 types are in lower case, whereas ASN.1 types are referred to with the first letter in upper case. Except as noted, all mappings are symmetrical.

#### 3.3.1. Boolean

Boolean is encoded as:

```
boolean = "TRUE" / "FALSE"
```

#### 3.3.2. NumericString

NumericString is encoded as:

```
numericstring = *DIGIT
```

#### 3.3.3. PrintableString

PrintableString is a restricted IA5String defined as:

```
printablestring = *( ps-char )
ps-restricted-char = 1DIGIT / 1ALPHA / " " / "'" / "+"
                  / ", " / "-" / "." / "/" / ":" / "=" / "?"
ps-delim          = "(" / ")"
ps-char           = ps-delim / ps-restricted-char
```

This can be used to represent real printable strings in EBNF.

#### 3.3.4. T.61String

In cases where T.61 strings are only used for conveying human interpreted information, the aim of a mapping is to render the characters appropriately in the remote character set, rather than to maximise reversibility. For these cases, the mappings to IA5 defined in CCITT Recommendation X.408 (1988) shall be used [CCITT/ISO88a]. These will then be encoded in ASCII.

There is also a need to represent Teletex Strings in ASCII, for some aspects of O/R Address. For these, the following encoding is used:

```
teletex-string    = *( ps-char / t61-encoded )
t61-encoded       = "{" 1* t61-encoded-char "}"
t61-encoded-char  = 3DIGIT
```

Common characters are mapped simply. Other octets are mapped using a quoting mechanism similar to the printable string mechanism. Each octet is represented as 3 decimal digits.

There are a number of places where a string may have a Teletex and/or Printable String representation. The following BNF is used to represent this.

```
teletex-and-or-ps = [ printablestring ] [ "*" teletex-string ]
```

The natural mapping is restricted to EBNF.ps-char, in order to make the full BNF easier to parse.

### 3.3.5. UTCTime

Both UTCTime and the RFC 822 822.date-time syntax contain: Year (lowest two digits), Month, Day of Month, hour, minute, second (optional), and Timezone. 822.date-time also contains an optional day of the week, but this is redundant. Therefore a symmetrical mapping can be made between these constructs.

Note:

In practice, a gateway will need to parse various illegal variants on 822.date-time. In cases where 822.date-time cannot be parsed, it is recommended that the derived UTCTime is set to the value at the time of translation.

When mapping to X.400, the UTCTime format which specifies the timezone offset shall be used.

When mapping to RFC 822, the 822.date-time format shall include a numeric timezone offset (e.g., +0000).

When mapping time values, the timezone shall be preserved as specified. The date shall not be normalised to any other timezone.

### 3.3.6. Integer

A basic ASN.1 Integer will be mapped onto EBNF.numericstring. In many cases ASN.1 will enumerate Integer values or use ENUMERATED. An EBNF encoding labelled-integer is provided. When mapping from EBNF to

ASN.1, only the integer value is mapped, and the associated text is discarded. When mapping from ASN.1 to EBNF, addition of an appropriate text label is strongly encouraged.

```
labelled-integer ::= [ key-string ] "(" numericstring ")"
```

```
key-string      = *key-char
```

```
key-char       = <a-z, A-Z, 0-9, and "-">
```

### 3.3.7. Object Identifier

Object identifiers are represented in a form similar to that given in ASN.1. The order is the same as for ASN.1 (big-endian). The numbers are mandatory, and used when mapping from the ASCII to ASN.1. The key-strings are optional. It is recommended that as many strings as possible are generated when mapping from ASN.1 to ASCII, to facilitate user recognition.

```
object-identifier ::= oid-comp object-identifier
                  | oid-comp
```

```
oid-comp ::= [ key-string ] "(" numericstring ")"
```

An example representation of an object identifier is:

```
joint-iso-ccitt(2) mhs (6) ipms (1) ep (11) ia5-text (0)
```

or

```
(2) (6) (1) (11) (0)
```

### 3.4. Encoding ASCII in Printable String

Some information in RFC 822 is represented in ASCII, and needs to be mapped into X.400 elements encoded as printable string. For this reason, a mechanism to represent ASCII encoded as PrintableString is needed.

A structured subset of EBNF.printablestring is now defined. This shall be used to encode ASCII in the PrintableString character set.

```

ps-encoded      = *( ps-restricted-char / ps-encoded-char )
ps-encoded-char = " (a) "           ; (@)
                  / " (p) "           ; (%)
                  / " (b) "           ; (!)
                  / " (q) "           ; (")
                  / " (u) "           ; ( _ )
                  / " (l) "           ; "( "
                  / " (r) "           ; ") "
                  / " (" 3DIGIT ")"

```

The 822.3DIGIT in EBNF.ps-encoded-char must have range 0-127, and is interpreted in decimal as the corresponding ASCII character. Special encodings are given for: at sign (@), percent (%), exclamation mark/bang (!), double quote ("), underscore (\_), left bracket ([), and right bracket (]). These characters, with the exception of round brackets, are not included in PrintableString, but are common in RFC 822 addresses. The abbreviations will ease specification of RFC 822 addresses from an X.400 system. These special encodings shall be interpreted in a case insensitive manner, but always generated in lower case.

A reversible mapping between PrintableString and ASCII can now be defined. The reversibility means that some values of printable string (containing round braces) cannot be generated from ASCII. Therefore, this mapping must only be used in cases where the printable strings may only be derived from ASCII (and will therefore have a restricted domain). For example, in this specification, it is only applied to a Domain Defined Attribute which will have been generated by use of this specification and a value such as "((" would not be possible.

To encode ASCII as PrintableString, the EBNF.ps-encoded syntax is used, with all EBNF.ps-restricted-char mapped directly. All other 822.CHAR are encoded as EBNF.ps-encoded-char.

To encode PrintableString as ASCII, parse PrintableString as EBNF.ps-encoded, and then reverse the previous mapping. If the PrintableString cannot be parsed, then the mapping is being applied in to an inappropriate value, and an error shall be given to the procedure doing the mapping. In some cases, it may be preferable to pass the printable string through unaltered.

Some examples are now given. Note the arrows which indicate asymmetrical mappings:

PrintableString		ASCII
'a demo.'	<->	'a demo.'
foo(a)bar	<->	foo@bar
(q) (u) (p) (q)	<->	"_%"
(a)	<->	@
(A)	->	@
(l)a(r)	<->	(a)
(126)	<->	~
(	->	(
(l)	<->	(

## Chapter 4 - Addressing

Addressing is probably the trickiest problem of an X.400 <-> RFC 822 gateway. Therefore it is given a separate chapter. This chapter, as a side effect, also defines a textual representation of an X.400 O/R Address.

Initially we consider an address in the (human) mail user sense of "what is typed at the mailsystem to reference a mail user". A basic RFC 822 address is defined by the EBNF EBNF.822-address:

```
822-address      = [ route ] addr-spec
```

In an 822-MTS protocol, the originator and each recipient are considered to be defined by such a construct. In an RFC 822 header, the EBNF.822-address is encapsulated in the 822.address syntax rule, and there may also be associated comments. None of this extra information has any semantics, other than to the end user.

The basic X.400 O/R Address, used by the MTS for routing, is defined by MTS.ORAddress. In IPMS, the MTS.ORAddress is encapsulated within IPMS.ORDescriptor.

It can be seen that RFC 822 822.address must be mapped with IPMS.ORDescriptor, and that RFC 822 EBNF.822-address must be mapped with MTS.ORAddress.

### 4.1. A textual representation of MTS.ORAddress

MTS.ORAddress is structured as a set of attribute value pairs. It is clearly necessary to be able to encode this in ASCII for gatewaying purposes. All components shall be encoded, in order to guarantee return of error messages, and to optimise third party replies.

## 4.2. Basic Representation

An O/R Address has a number of structured and unstructured attributes. For each unstructured attribute, a key and an encoding is specified. For structured attributes, the X.400 attribute is mapped onto one or more attribute value pairs. For domain defined attributes, each element of the sequence will be mapped onto a triple (key and two values), with each value having the same encoding. The attributes are as follows, with 1984 attributes given in the first part of the table. For each attribute, a reference is given, consisting of the relevant sections in X.402 / ISO 10021-2, and the extension identifier for 88 only attributes:

Attribute (Component)	Key	Enc	Ref	Id
84/88 Attributes				
MTS.CountryName	C	P	18.3.3	
MTS.AdministrationDomainName	ADMD	P	18.3.1	
MTS.PrivateDomainName	PRMD	P	18.3.21	
MTS.NetworkAddress	X121	N	18.3.7	
MTS.TerminalIdentifier	T-ID	P	18.3.23	
MTS.OrganizationName	O	P/T	18.3.9	
MTS.OrganizationalUnitNames.value	OU	P/T	18.3.10	
MTS.NumericUserIdentifier	UA-ID	N	18.3.8	
MTS.PersonalName	PN	P/T	18.3.12	
MTS.PersonalName.surname	S	P/T	18.3.12	
MTS.PersonalName.given-name	G	P/T	18.3.12	
MTS.PersonalName.initials	I	P/T	18.3.12	
MTS.PersonalName				
.generation-qualifier	GQ	P/T	18.3.12	
MTS.DomainDefinedAttribute.value	DD	P/T	18.1	
88 Attributes				
MTS.CommonName	CN	P/T	18.3.2	1
MTS.TeletexCommonName	CN	P/T	18.3.2	2
MTS.TeletexOrganizationName	O	P/T	18.3.9	3
MTS.TeletexPersonalName	PN	P/T	18.3.12	4
MTS.TeletexPersonalName.surname	S	P/T	18.3.12	4
MTS.TeletexPersonalName.given-name	G	P/T	18.3.12	4
MTS.TeletexPersonalName.initials	I	P/T	18.3.12	4
MTS.TeletexPersonalName				
.generation-qualifier	GQ	P/T	18.3.12	4
MTS.TeletexOrganizationalUnitNames				
.value	OU	P/T	18.3.10	5
MTS.TeletexDomainDefinedAttribute				
.value	DD	P/T	18.1	6

MTS.PDSName	PD-SERVICE	P	18.3.11	7
MTS.PhysicalDeliveryCountryName	PD-C	P	18.3.13	8
MTS.PostalCode	PD-CODE	P	18.3.19	9
MTS.PhysicalDeliveryOfficeName	PD-OFFICE	P/T	18.3.14	10
MTS.PhysicalDeliveryOfficeNumber	PD-OFFICE-NUM	P/T	18.3.15	11
MTS.ExtensionORAddressComponents	PD-EXT-ADDRESS	P/T	18.3.4	12
MTS.PhysicalDeliveryPersonName	PD-PN	P/T	18.3.17	13
MTS.PhysicalDeliveryOrganizationName	PD-O	P/T	18.3.16	14
MTS.ExtensionPhysicalDelivery AddressComponents	PD-EXT-DELIVERY	P/T	18.3.5	15
MTS.UnformattedPostalAddress	PD-ADDRESS	P/T	18.3.25	16
MTS.StreetAddress	PD-STREET	P/T	18.3.22	17
MTS.PostOfficeBoxAddress	PD-BOX	P/T	18.3.18	18
MTS.PosteRestanteAddress	PD-RESTANTE	P/T	18.3.20	19
MTS.UniquePostalName	PD-UNIQUE	P/T	18.3.26	20
MTS.LocalPostalAttributes	PD-LOCAL	P/T	18.3.6	21
MTS.ExtendedNetworkAddress .e163-4-address.number	NET-NUM	N	18.3.7	22
MTS.ExtendedNetworkAddress .e163-4-address.sub-address	NET-SUB	N	18.3.7	22
MTS.ExtendedNetworkAddress .psap-address	NET-PSAP	X	18.3.7	22
MTS.TerminalType	T-TY	I	18.3.24	23

The following keys identify different EBNF encodings, which are associated with the ASCII representation of MTS.ORAddress.

Key	Encoding
P	printablestring
N	numericstring
T	teletex-string
P/T	teletex-and-or-ps
I	labelled-integer
X	presentation-address

The BNF for presentation-address is taken from the specification "A String Encoding of Presentation Address" [Kille89a].

In most cases, the EBNF encoding maps directly to the ASN.1 encoding of the attribute. There are a few exceptions. In cases where an attribute can be encoded as either a PrintableString or NumericString (Country, ADMD, PRMD), either form is mapped into the BNF. When generating ASN.1, the NumericString encoding shall be used if the string contains only digits.

There are a number of cases where the P/T (teletex-and-or-ps) representation is used. Where the key maps to a single attribute,



this choice is reflected in the encoding of the attribute (attributes 10-21). For most of the 1984 attributes and common name, there is a printablestring and a teletex variant. This pair of attributes is mapped onto the single component here. This will give a clean mapping for the common cases where only one form of the name is used.

Recently, ISO has undertaken work to specify a string form of O/R Address [CCITT/ISO91a]. This has specified a number of string keywords for attributes. As RFC 1148 was an input to this work, many of the keywords are the same. To increase compatability, the following alternative values shall be recognised when mapping from RFC 822 to X.400. These shall not be generated when mapping from X.400 to RFC 822.

Keyword	Alternative
ADMD	A
PRMD	P
GQ	Q
X121	X.121
UA-ID	N-ID
PD-OFFICE-NUMBER	PD-OFFICE NUMBER

When mapping from RFC 822 to X.400, the keywords: OU1, OU2, OU3, and OU4, shall be recognised. If these are present, no keyword OU shall be present. These will be treated as ordered values of OU.

#### 4.2.1. Encoding of Personal Name

Handling of Personal Name and Teletex Personal Name based purely on the EBNF.standard-type syntax defined above is likely to be clumsy. It seems desirable to utilise the "human" conventions for encoding these components. A syntax is defined, which is designed to provide a clean encoding for the common cases of O/R Address specification where:

1. There is no generational qualifier
2. Initials contain only letters
3. Given Name does not contain full stop ("."), and is at least two characters long.
4. Surname does not contain full stop in the first two characters.
5. If Surname is the only component, it does not contain full stop.

The following EBNF is defined:

```

encoded-pn      = [ given "." ] *( initial "." ) surname
given           = 2*<ps-char not including ".">
initial         = ALPHA
surname         = printablestring

```

This is used to map from any string containing only printable string characters to an O/R address personal name. To map from a string to O/R Address components, parse the string according to the EBNF. The given name and surname are assigned directly. All EBNF.initial tokens are concatenated without intervening full stops to generate the initials component.

For an O/R address which follows the above restrictions, a string is derived in the natural manner. In this case, the mapping will be reversible.

For example:

```

GivenName      = "Marshall"
Surname        = "Rose"

```

Maps with "Marshall.Rose"

```

Initials       = "MT"
Surname        = "Rose"

```

Maps with "M.T.Rose"

```

GivenName      = "Marshall"
Initials       = "MT"
Surname        = "Rose"

```

Maps with "Marshall.M.T.Rose"

Note that X.400 suggest that Initials is used to encode ALL initials. Therefore, the defined encoding is "natural" when either GivenName or Initials, but not both, are present. The case where both are present can be encoded, but this appears to be contrived!

#### 4.2.2. Standard Encoding of MTS.ORAddress

Given this structure, we can specify a BNF representation of an O/R Address.

```

std-or-address = 1*( "/" attribute "=" value ) "/"
attribute      = standard-type
                / "RFC-822"
                / registered-dd-type
                / dd-key "." std-printablestring
standard-type  = key-string

registered-dd-type
                = key-string
dd-key         = key-string

value          = std-printablestring

std-printablestring
                = *( std-char / std-pair )
std-char       = <"{", "}", "*", and any ps-char
                except "/" and "=">
std-pair       = "$" ps-char

```

The standard-type is any key defined in the table in Section 4.2, except PN, and DD. The BNF leads to a set of attribute/value pairs. The value is interpreted according to the EBNF encoding defined in the table.

If the standard-type is PN, the value is interpreted according to EBNF.encoded-pn, and the components of MTS.PersonalName and/or MTS.TeletexPersonalName derived accordingly.

If dd-key is the recognised Domain Defined string (DD), then the type and value are interpreted according to the syntax implied from the encoding, and aligned to either the teletex or printable string form. Key and value shall have the same encoding.

If value is "RFC-822", then the (printable string) Domain Defined Type of "RFC-822" is assumed. This is an optimised encoding of the domain defined type defined by this specification.

The matching of all keywords shall be done in a case-independent manner.

EBNF.std-or-address uses the characters "/" and "=" as delimiters. Domain Defined Attributes and any value may contain these characters. A quoting mechanism, using the non-printable string "\$" is used to allow these characters to be represented.

If the value is registered-dd-type, and the value is registered at the Internet Assigned Numbers Authority (IANA) as an accepted Domain Defined Attribute type, then the value shall be interpreted

accordingly. This restriction maximises the syntax checking which can be done at a gateway.

#### 4.3. EBNF.822-address <-> MTS.ORAddress

Ideally, the mapping specified would be entirely symmetrical and global, to enable addresses to be referred to transparently in the remote system, with the choice of gateway being left to the Message Transfer Service. There are two fundamental reasons why this is not possible:

1. The syntaxes are sufficiently different to make this awkward.
2. In the general case, there would not be the necessary administrative co-operation between the X.400 and RFC 822 worlds, which would be needed for this to work.

Therefore, an asymmetrical mapping is defined, which can be symmetrical where there is appropriate administrative control.

##### 4.3.1. X.400 encoded in RFC 822

The std-or-address syntax is used to encode O/R Address information in the 822.local-part of EBNF.822-address. In some cases, further O/R Address information is associated with the 822.domain component. This cannot be used in the general case, due to character set problems, and to the variants of X.400 O/R Addresses which use different attribute types. The only way to encode the full PrintableString character set in a domain is by use of the 822.domain-ref syntax (i.e. 822.atom). This is likely to cause problems on many systems. The effective character set of domains is in practice reduced from the RFC 822 set, by restrictions imposed by domain conventions and policy, and by restrictions in RFC 821.

A generic 822.address consists of a 822.local-part and a sequence of 822.domains (e.g., <@domain1,@domain2:user@domain3>). All except the 822.domain associated with the 822.local-part (domain3 in this case) are considered to specify routing within the RFC 822 world, and will not be interpreted by the gateway (although they may have identified the gateway from within the RFC 822 world).

The 822.domain associated with the 822.local-part identifies the gateway from within the RFC 822 world. This final 822.domain may be used to determine some number of O/R Address attributes, where this does not conflict with the first role. RFC 822 routing to gateways will usually be set up to facilitate the 822.domain being used for both purposes. The following O/R Address attributes are considered

as a hierarchy, and may be specified by the domain. They are (in order of hierarchy):

Country, ADMD, PRMD, Organisation, Organisational Unit

There may be multiple Organisational Units.

A global mapping is defined between domain specifications, and some set of attributes. This association proceeds hierarchically. For example, if a domain implies ADMD, it also implies country. Subdomains under this are associated according to the O/R Address hierarchy. For example:

=> "AC.UK" might be associated with  
C="GB", ADMD="GOLD 400", PRMD="UK.AC"

then domain "R-D.Salford.AC.UK" maps with  
C="GB", ADMD="GOLD 400", PRMD="UK.AC", O="Salford", OU="R-D"

There are three basic reasons why a domain/attribute mapping might be maintained, as opposed to using simply subdomains:

1. As a shorthand to avoid redundant X.400 information. In particular, there will often be only one ADMD per country, and so it does not need to be given explicitly.
2. To deal with cases where attribute values do not fit the syntax:

```
domain-syntax    = alphanum [ *alphanumhyphen alphanum ]
alphanum         = <ALPHA or DIGIT>
alphanumhyphen   = <ALPHA or DIGIT or HYPHEN>
```

Although RFC 822 allows for a more general syntax, this restricted syntax is chosen as it is the one chosen by the various domain service administrations.

3. To deal with missing elements in the hierarchy. A domain may be associated with an omitted attribute in conjunction with several present ones. When performing the algorithmic insertion of components lower in the hierarchy, the omitted value shall be skipped. For example, if "HNE.EGM" is associated with "C=TC", "ADMD=ECQ", "PRMD=HNE", and omitted organisation, then "ZI.HNE.EGM" is mapped with "C=TC", "ADMD=ECQ", "PRMD=HNE", "OU=ZI". Attributes may have null values, and this is treated separately from omitted attributes (whilst it would be bad practice to treat these

two cases differently, they must be allowed for).

This set of mappings needs be known by the gateways relaying between the RFC 822 world, and the O/R Address space associated with the mapping in question. There needs to be a single global definition of this set of mappings. A mapping implies an administrative equivalence between the two parts of the namespaces which are mapped together. To correctly route in all cases, it is necessary for all gateways to know the mapping. To facilitate distribution of a global set of mappings, a format for the exchange of this information is defined in Appendix F.

The remaining attributes are encoded on the LHS, using the EBNF.std-or-address syntax. For example:

```
/I=J/S=Linnimouth/GQ=5/@Marketing.Widget.COM
```

encodes the MTS.ORAddress consisting of:

MTS.CountryName	= "TC"
MTS.AdministrationDomainName	= "BT"
MTS.OrganizationName	= "Widget"
MTS.OrganizationalUnitNames.value	= "Marketing"
MTS.PersonalName.surname	= "Linnimouth"
MTS.PersonalName.initials	= "J"
MTS.PersonalName.generation-qualifier	= "5"

The first three attributes are determined by the domain Widget.COM. Then, the first element of OrganizationalUnitNames is determined systematically, and the remaining attributes are encoded on the LHS. In an extreme case, all of the attributes will be on the LHS. As the domain cannot be null, the RHS will simply be a domain indicating the gateway.

The RHS (domain) encoding is designed to deal cleanly with common addresses, and so the amount of information on the RHS is maximised. In particular, it covers the Mnemonic O/R Address using a 1984 compatible encoding. This is seen as the dominant form of O/R Address. Use of other forms of O/R Address, and teletex encoded attributes will require an LHS encoding.

There is a further mechanism to simplify the encoding of common cases, where the only attributes to be encoded on the LHS is a (non-Teletex) Personal Name attributes which comply with the restrictions of 4.2.1. To achieve this, the 822.local-part shall be encoded as EBNF.encoded-pn. In the previous example, if the GenerationQualifier was not present in the previous example O/R Address, it would map with the RFC 822 address: J.Linnimouth@Marketing.Widget.COM.

From the standpoint of the RFC 822 Message Transfer System, the domain specification is simply used to route the message in the standard manner. The standard domain mechanisms are used to select appropriate gateways for the corresponding O/R Address space. In most cases, this will be done by registering the higher levels, and assuming that the gateway can handle the lower levels.

#### 4.3.2. RFC 822 encoded in X.400

In some cases, the encoding defined above may be reversed, to give a "natural" encoding of genuine RFC 822 addresses. This depends largely on the allocation of appropriate management domains.

The general case is mapped by use of domain defined attributes. A Domain defined type "RFC-822" is defined. The associated attribute value is an ASCII string encoded according to Section 3.3.3 of this specification. The interpretation of the ASCII string depends on the context of the gateway.

1. In the context of RFC 822, and RFC 920 [Crocker82a,Postel84a], the string can be used directly.
2. In the context of the JNT Mail protocol, and the NRS [Kille84a,Larmouth83a], the string shall be interpreted according to Mailgroup Note 15 [Kille84b].
3. In the context of UUCP based systems, the string shall be interpreted as defined in [Horton86a].

Other O/R Address attributes will be used to identify a context in which the O/R Address will be interpreted. This might be a Management Domain, or some part of a Management Domain which identifies a gateway MTA. For example:

C	= "GB"
ADMD	= "GOLD 400"
PRMD	= "UK.AC"
O	= "UCL"
OU	= "CS"
"RFC-822"	= "Jimmy(a)WIDGET-LABS.CO.UK"

OR

C	= "TC"
ADMD	= "Wizz.mail"
PRMD	= "42"
"rfc-822"	= "postel(a)venera.isi.edu"

Note in each case the PrintableString encoding of "@" as "(a)". In the second example, the "RFC-822" domain defined attribute is interpreted everywhere within the (Private) Management Domain. In the first example, further attributes are needed within the Management Domain to identify a gateway. Thus, this scheme can be used with varying levels of Management Domain co-operation.

There is a limit of 128 characters in the length of value of a domain defined attribute, and an O/R Address can have a maximum of four domain defined attributes. Where the printable string generated from the RFC 822 address exceeds this value, additional domain defined attributes are used to enable up to 512 characters to be encoded. These attributes shall be filled completely before the next one is started. The DDA keywords are: RFC822C1; RFC822C2; RFC822C3. Longer addresses cannot be encoded.

There is, analagous with 4.3.1, a means to associate parts of the O/R Address hierarchy with domains. There is an analogous global mapping, which in most cases will be the inverse of the domain to O/R address mapping. The mapping is maintained separately, as there may be differences (e.g., two alternate domain names map to the same set of O/R address components).

#### 4.3.3. Component Ordering

In most cases, ordering of O/R Address components is not significant for the mappings specified. However, Organisational Units (printable string and teletex forms) and Domain Defined Attributes are specified as SEQUENCE in MTS.ORAddress, and so their order may be significant. This specification needs to take account of this:

1. To allow consistent mapping into the domain hierarchy
2. To ensure preservation of order over multiple mappings.

There are three places where an order is specified:

1. The text encoding (std-or-address) of MTS.ORAddress as used in the local-part of an RFC 822 address. An order is needed for those components which may have multiple values (Organisational Unit, and Domain Defined Attributes). When generating an 822.std-or-address, components of a given type shall be in hierarchical order with the most significant component on the RHS. If there is an Organisation Attribute, it shall be to the right of any Organisational Unit attributes. These requirements are for the following reasons:



- Alignment to the hierarchy of other components in RFC 822 addresses (thus, Organisational Units will appear in the same order, whether encoded on the RHS or LHS). Note the differences of JNT Mail as described in Appendix B.
- Backwards compatibility with RFC 987/1026.
- To ensure that gateways generate consistent addresses. This is both to help end users, and to generate identical message ids.

Further, it is recommended that all other attributes are generated according to this ordering, so that all attributes so encoded follow a consistent hierarchy. When generating 822.msg-id, this order shall be followed.

2. For the Organisational Units (OU) in MTS.ORAddress, the first OU in the SEQUENCE is the most significant, as specified in X.400.
3. For the Domain Defined Attributes in MTS.ORAddress, the First Domain Defined Attribute in the SEQUENCE is the most significant.

Note that although this ordering is mandatory for this mapping, there are NO implications on ordering significance within X.400, where this is a Management Domain issue.

#### 4.3.4. RFC 822 -> X.400

There are two basic cases:

1. X.400 addresses encoded in RFC 822. This will also include RFC 822 addresses which are given reversible encodings.
2. "Genuine" RFC 822 addresses.

The mapping shall proceed as follows, by first assuming case 1).

##### STAGE I.

1. If the 822-address is not of the form:

local-part "@" domain

take the domain which will be routed on and apply step 2 of stage 1 to derive (a possibly null) set of attributes. Then

go to stage II.

NOTE: It may be appropriate to reduce a source route address to this form by removal of all but the last domain. In terms of the design intentions of RFC 822, this would be an incorrect action. However, in most real cases, it will do the "right" thing and provide a better service to the end user. This is a reflection on the excessive and inappropriate use of source routing in RFC 822 based systems. Either approach, or the intermediate approach of stripping only domain references which reference the local gateway are conformant to this specification.

2. Attempt to parse EBNF.domain as:

```
*( domain-syntax "." ) known-domain
```

Where EBNF.known-domain is the longest possible match in the set of globally defined mappings (see Appendix F). If this fails, and the EBNF.domain does not explicitly identify the local gateway, go to stage II. If the domain explicitly identifies the gateway, allocate no attributes. Otherwise, allocate the attributes associated with EBNF.known-domain. For each component, systematically allocate the attribute implied by each EBNF.domain-syntax component in the order: C, ADMD, PRMD, O, OU. Note that if the mapping used identifies an "omitted attribute", then this attribute should be omitted in the systematic allocation. If this new component exceeds an upper bound (ADMD: 16; PRMD: 16; O: 64; OU: 32) or it would lead to more than four OUs, then go to stage II with the attributes derived.

At this stage, a set of attributes has been derived, which will give appropriate routing within X.400. If any of the later steps of Stage I force use of Stage II, then these attributes should be used in Stage II.

3. If the 822.local-part uses the 822.quoted-string encoding, remove this quoting. If this unquoted 822.local-part has leading space, trailing space, or two adjacent spaces go to stage II.
4. If the unquoted 822.local-part contains any characters not in PrintableString, go to stage II.
5. Parse the (unquoted) 822.local-part according to the EBNF EBNF.std-or-address. Checking of upper bounds should not be

done at this point. If this parse fails, parse the local-part according to the EBNF EBNF.encoded-pn. If this parse fails, go to stage II. The result is a set of type/value pairs. If the set of attributes leads to an address of any form other than mnemonic form, then only these attributes should be taken. If (for mnemonic form) the values generated conflict with those derived in step 2 (e.g., a duplicated country attribute), the domain is assumed to be a remote gateway. In this case, take only the LHS derived attributes, together with any RHS derived attributes which are more significant than the most significant attribute which is duplicated (e.g., if there is a duplicate PRMD, but no LHS derived ADMD and country, then the ADMD and country should be taken from the RHS). otherwise add LHS and RHS derived attributes together.

6. Associate the EBNF.attribute-value syntax (determined from the identified type) with each value, and check that it conforms. If not, go to stage II.
7. Ensure that the set of attributes conforms both to the MTS.ORAddress specification and to the restrictions on this set given in X.400, and that no upper bounds are exceeded for any attribute. If not go to stage II.
8. Build the O/R Address from this information.

#### STAGE II.

This will only be reached if the RFC 822 EBNF.822-address is not a valid X.400 encoding. This implies that the address must refer to a recipient on an RFC 822 system. Such addresses shall be encoded in an X.400 O/R Address using a domain defined attribute.

1. Convert the EBNF.822-address to PrintableString, as specified in Chapter 3.
2. Generate the "RFC-822" domain defined attribute from this string.
3. Build the rest of the O/R Address in the manner described below.

It may not be possible to encode the domain defined attribute due to length restrictions. If the limit is exceeded by a mapping at the MTS level, then the gateway shall reject the message in question. If this occurs at the IPMS level, then the action will depend on the policy being taken for IPMS encoding, which is discussed in Section

### 5.1.3.

If Stage I has identified a set of attributes, use these to build the remainder of the address. The administrative equivalence of the mappings will ensure correct routing through X.400 to a gateway back to RFC 822.

If Stage I has not identified a set of attributes, the remainder of the O/R address effectively identifies a source route to a gateway from the X.400 side. There are three cases, which are handled differently:

#### 822-MTS Return Address

This shall be set up so that errors are returned through the same gateway. Therefore, the O/R Address of the local gateway shall be used.

#### IPMS Addresses

These are optimised for replying. In general, the message may end up anywhere within the X.400 world, and so this optimisation identifies a gateway appropriate for the RFC 822 address being converted. The 822.domain to which the address would be routed is used to select an appropriate gateway. A globally defined set of mappings is used, which identifies (the O/R Address components of) appropriate gateways for parts of the domain namespace. The longest possible match on the 822.domain defines which gateway to use. The table format for distribution of this information is defined in Appendix F.

This global mapping is used for parts of the RFC 822 namespace which do not have an administrative equivalence with any part of the X.400 namespace, but for which it is desirable to identify a preferred X.400 gateway in order to optimise routing.

If no mapping is found for the 822.domain, a default value (typically that of the local gateway) is used. It is never appropriate to ignore the globally defined mappings. In some cases, it may be appropriate to locally override the globally defined mappings (e.g., to identify a gateway close to a recipient of the message). This is likely to be where the global mapping identifies a public gateway, and the local gateway has an agreement with a private gateway which it prefers to use.

#### 822-MTS Recipient

As the RFC 822 and X.400 worlds are fully connected, there

is no technical reason for this situation to occur. In some cases, routing may be configured to connect two parts of the RFC 822 world using X.400. The information that this part of the domain space should be routed by X.400 rather than remaining within the RFC 822 world will be configured privately into the gateway in question. The O/R address shall then be generated in the same manner as for an IPMS address, using the globally defined mappings. It is to support this case that the definition of the global domain to gateway mapping is important, as the use of this mapping will lead to a remote X.400 address, which can be routed by X.400 routing procedures. The information in this mapping shall not be used as a basis for deciding to convert a message from RFC 822 to X.400.

#### 4.3.4.1. Heuristics for mapping RFC 822 to X.400

RFC 822 users will often use an LHS encoded address to identify an X.400 recipient. Because the syntax is fairly complex, a number of heuristics may be applied to facilitate this form of usage. A gateway should take care not to be overly "clever" with heuristics, as this may cause more confusion than a more mechanical approach. The heuristics are as follows:

1. Ignore the omission of a trailing "/" in the std-or syntax.
2. If there is no ADMD component, and both country and PRMD are present, the value of /ADMD= / (single space) is assumed.
3. Parse the unquoted local part according to the EBNF colon-or-address. This may facilitate users used to this delimiter.

colon-or-address = 1\*(attribute "=" value ";" \*(LWSP-char))

The remaining heuristic relates to ordering of address components. The ordering of attributes may be inverted or mixed. For this reason, the following heuristics may be applied:

4. If there is an Organisation attribute to the left of any Org Unit attribute, assume that the hierarchy is inverted.

#### 4.3.5. X.400 -> RFC 822

There are two basic cases:

1. RFC 822 addresses encoded in X.400.

2. "Genuine" X.400 addresses. This may include symmetrically encoded RFC 822 addresses.

When a MTS Recipient O/R Address is interpreted, gatewaying will be selected if there is a single "RFC-822" domain defined attribute present and the local gateway is identified by the remainder of the O/R Address. In this case, use mapping A. For other O/R Addresses which

1. Contain the special attribute.

AND

2. Identifies the local gateway or any other known gateway with the other attributes.

use mapping A. In other cases, use mapping B.

NOTE:

A pragmatic approach would be to assume that any O/R Address with the special domain defined attribute identifies an RFC 822 address. This will usually work correctly, but is in principle not correct. Use of this approach is conformant to this specification.

Mapping A

1. Map the domain defined attribute value to ASCII, as defined in Chapter 3.

Mapping B

This is used for X.400 addresses which do not use the explicit RFC 822 encoding.

1. For all string encoded attributes, remove any leading or trailing spaces, and replace adjacent spaces with a single space.

The only attribute which is permitted to have zero length is the ADMD. This should be mapped onto a single space.

These transformations are for lookup only. If an EBNF.std-or-address mapping is used as in 4), then the original values should be used.

2. Map numeric country codes to the two letter values.

3. Noting the hierarchy specified in 4.3.1 and including omitted attributes, determine the maximum set of attributes which have an associated domain specification in the globally defined mapping. If no match is found, allocate the domain as the domain specification of the local gateway, and go to step 5.

Note: It might be appropriate to use a non-local domain. This would be selected by a global mapping analagous to the one described at the end of 4.3.4. This is not done, primarily because use of RFC 822 to connect X.400 systems is not expected to be significant.

In cases where the address refers to an X.400 UA, it is important that the generated domain will correctly route to a gateway. In general, this is achieved by carefully co-ordinating RFC 822 routing with the definition of the global mappings, as there is no easy way for the gateway to make this check. One rule that shall be used is that domains with only one component will not route to a gateway. If the generated domain does not route correctly, the address is treated as if no match is found.

4. The mapping identified in 3) gives a domain, and an O/R address prefix. Follow the hierarchy: C, ADMD, PRMD, O, OU. For each successive component below the O/R address prefix, which conforms to the syntax EBNF.domain-syntax (as defined in 4.3.1), allocate the next subdomain. At least one attribute of the X.400 address shall not be mapped onto subdomain, as 822.local-part cannot be null. If there are omitted attributes in the O/R address prefix, these will have correctly and uniquely mapped to a domain component. Where there is an attribute omitted below the prefix, all attributes remaining in the O/R address shall be encoded on the LHS. This is to ensure a reversible mapping. For example, if there is an address /S=XX/O=YY/ADMD=A/C=NN/ and a mapping for /ADMD=A/C=NN/ is used, then /S=XX/O=YY/ is encoded on the LHS.
5. If the address is not mnemonic form (form 1 variant 1), then all of the attributes in the address should be encoded on the LHS in EBNF.std-or-address syntax, as described below.

For addresses of mnemonic form, if the remaining components are personal-name components, conforming to the restrictions of 4.2.1, then EBNF.encoded-pn is derived to form 822.local-part. In other cases the remaining components are

simply encoded as 822.local-part using the EBNF.std-or-address syntax. If necessary, the 822.quoted-string encoding is used. The following are examples of legal quoting: "a b".c@x; "a b.c"@x. Either form may be generated, but the latter is preferred.

If the derived 822.local-part can only be encoded by use of 822.quoted-string, then use of the mapping defined in [Kille89b] may be appropriate. Use of this mapping is discouraged.

#### 4.4. Repeated Mappings

There are two types of repeated mapping:

1. A recursive mapping, where the repeat is within one gateway
2. A source route, where the repetition occurs across multiple gateways

##### 4.4.1. Recursive Mappings

It is possible to supply an address which is recursive at a single gateway. For example:

C	=	"XX"
ADMD	=	"YY"
O	=	"ZZ"
"RFC-822"	=	"Smith(a)ZZ.YY.XX"

This is mapped first to an RFC 822 address, and then back to the X.400 address:

C	=	"XX"
ADMD	=	"YY"
O	=	"ZZ"
Surname	=	"Smith"

In some situations this type of recursion may be frequent. It is important that where this occurs, that no unnecessary protocol conversion occurs. This will minimise loss of service.

##### 4.4.2. Source Routes

The mappings defined are symmetrical and reversible across a single gateway. The symmetry is particularly useful in cases of (mail exploder type) distribution list expansion. For example, an X.400 user sends to a list on an RFC 822 system which he belongs to. The



received message will have the originator and any 3rd party X.400 O/R Addresses in correct format (rather than doubly encoded). In cases (X.400 or RFC 822) where there is common agreement on gateway identification, then this will apply to multiple gateways.

When a message traverses multiple gateways, the mapping will always be reversible, in that a reply can be generated which will correctly reverse the path. In many cases, the mapping will also be symmetrical, which will appear clean to the end user. For example, if countries "AB" and "XY" have RFC 822 networks, but are interconnected by X.400, the following may happen: The originator specifies:

Joe.Soap@Widget.PTT.XY

This is routed to a gateway, which generates:

C	= "XY"
ADMD	= "PTT"
PRMD	= "Griddle MHS Providers"
Organisation	= "Widget Corporation"
Surname	= "Soap"
Given Name	= "Joe"

This is then routed to another gateway where the mapping is reversed to give:

Joe.Soap@Widget.PTT.XY

Here, use of the gateway is transparent.

Mappings will only be symmetrical where mapping tables are defined. In other cases, the reversibility is more important, due to the (far too frequent) cases where RFC 822 and X.400 services are partitioned.

The syntax may be used to source route. THIS IS STRONGLY DISCOURAGED. For example:

X.400 -> RFC 822 -> X.400

C	= "UK"
ADMD	= "Gold 400"
PRMD	= "UK.AC"
"RFC-822"	= "/PN=Dupal/DD.Title=Manager/(a)Inria.ATLAS.FR"

This will be sent to an arbitrary UK Academic Community gateway by X.400. Then it will be sent by JNT Mail to another gateway determined by the domain Inria.ATLAS.FR (FR.ATLAS.Inria). This will

then derive the X.400 O/R Address:

C	= "FR"
ADMD	= "ATLAS"
PRMD	= "Inria"
PN.S	= "Duval"
"Title"	= "Manager"

Similarly:

RFC 822 -> X.400 -> RFC 822

`"/C=UK/ADMD=BT/PRMD=AC/RFC-822=jj(a)seismo.css.gov/"@monet.berkeley.edu`

This will be sent to monet.berkeley.edu by RFC 822, then to the AC PRMD by X.400, and then to jj@seismo.css.gov by RFC 822.

#### 4.5. Directory Names

Directory Names are an optional part of O/R Name, along with O/R Address. The RFC 822 addresses are mapped onto the O/R Address component. As there is no functional mapping for the Directory Name on the RFC 822 side, a textual mapping is used. There is no requirement for reversibility in terms of the goals of this specification. There may be some loss of functionality in terms of third party recipients where only a directory name is given, but this seems preferable to the significant extra complexity of adding a full mapping for Directory Names.

Note: There is ongoing work on specification of a "user friendly" format for directory names. If this is adopted as an internet standard, it will be recommended, but not required, for use here.

#### 4.6. MTS Mappings

The basic mappings at the MTS level are:

- 1) 822-MTS originator ->
 

MTS.PerMessageSubmissionFields.originator-name
MTS.OtherMessageDeliveryFields.originator-name ->
822-MTS originator
- 2) 822-MTS recipient ->
 

MTS.PerRecipientMessageSubmissionFields
MTS.OtherMessageDeliveryFields.this-recipient-name ->
822-MTS recipient

822-MTS recipients and return addresses are encoded as EBNF.822-

address.

The MTS Originator is always encoded as MTS.OriginatorName, which maps onto MTS.ORAddressAndOptionalDirectoryName, which in turn maps onto MTS.ORName.

#### 4.6.1. RFC 822 -> X.400

From the 822-MTS Originator, use the basic ORAddress mapping, to generate MTS.PerMessageSubmissionFields.originator-name (MTS.ORName), without a DirectoryName.

For recipients, the following settings are made for each component of MTS.PerRecipientMessageSubmissionFields.

recipient-name

This is derived from the 822-MTS recipient by the basic ORAddress mapping.

originator-report-request

This is to be set according to content return policy, as discussed in Section 5.2.

explicit-conversion

This optional component is omitted, as this service is not needed

extensions

The default value (no extensions) is used

#### 4.6.2. X.400 -> RFC 822

The basic functionality is to generate the 822-MTS originator and recipients. There is information present on the X.400 side, which cannot be mapped into analogous 822-MTS services. For this reason, new RFC 822 fields are added for the MTS Originator and Recipients. The information discarded at the 822-MTS level will be present in these fields. In some cases a (positive) delivery report will be generated.

##### 4.6.2.1. 822-MTS Mappings

Use the basic ORAddress mapping, to generate the 822-MTS originator (return address) from MTS.OtherMessageDeliveryFields.originator-name (MTS.ORName). If MTS.ORName.directory-name is present, it is discarded. (Note that it will be presented to the user, as described in 4.6.2.2).

The 822-MTS recipient is conceptually generated from `MTS.OtherMessageDeliveryFields.this-recipient-name`. This is done by taking `MTS.OtherMessageDeliveryFields.this-recipient-name`, and generating an 822-MTS recipient according to the basic ORAddress mapping, discarding `MTS.ORName.directory-name` if present. However, if this model was followed exactly, there would be no possibility to have multiple 822-MTS recipients on a single message. This is unacceptable, and so layering is violated. The mapping needs to use the MTA level information, and map each value of `MTA.PerRecipientMessageTransferFields.recipient-name`, where the responsibility bit is set, onto an 822-MTS recipient.

#### 4.6.2.2. Generation of RFC 822 Headers

Not all per-recipient information can be passed at the 822-MTS level. For this reason, two new RFC 822 headers are created, in order to carry this information to the RFC 822 recipient. These fields are `"X400-Originator:"` and `"X400-Recipients:"`.

The `"X400-Originator:"` field is set to the same value as the 822-MTS originator. In addition, if `MTS.OtherMessageDeliveryFields.originator-name` (`MTS.ORName`) contains `MTS.ORName.directory-name` then this Directory Name shall be represented in an `822.comment`.

Recipient names, taken from each value of `MTS.OtherMessageDeliveryFields.this-recipient-name` and `MTS.OtherMessageDeliveryFields.other-recipient-names` are made available to the RFC 822 user by use of the `"X400-Recipients:"` field. By taking the recipients at the MTS level, disclosure of recipients will be dealt with correctly. However, this conflicts with a desire to optimise mail transfer. There is no problem when disclosure of recipients is allowed. Similarly, there is no problem if there is only one RFC 822 recipient, as the `"X400-Recipients"` field is only given one address.

There is a problem if there are multiple RFC 822 recipients, and disclosure of recipients is prohibited. Two options are allowed:

1. Generate one copy of the message for each RFC 822 recipient, with the `"X400-Recipients"` field correctly set to the recipient of that copy. This is functionally correct, but is likely to be more expensive.
2. Discard the per-recipient information, and insert a field:

`X400-Recipients: non-disclosure;;`

This is the recommended option.

A third option of ignoring the disclosure flag is not allowed. If any MTS.ORName.directory-name is present, it shall be represented in an 822.comment.

If MTS.OtherMessageDeliveryFields.orignally-intended-recipient-name is present, then there has been redirection, or there has been distribution list expansion. Distribution list expansion is a per-message option, and the information associated with this is represented by the "DL-Expansion-History:" field described in Section 5.3.6. Other information is represented in an 822.comment associated with MTS.OtherMessageDeliveryFields.this-recipient-name. The message may be delivered to different RFC 822 recipients, and so several addresses in the "X400-Recipients:" field may have such comments. The non-commented recipient is the RFC 822 recipient. The EBNF of the comment is:

```

redirect-comment =
    [ "Originally To:" ] mailbox "Redirected"
    [ "Again" ] "on" date-time
    "To:" redirection-reason

redirection-reason =
    "Recipient Assigned Alternate Recipient"
    / "Originator Requested Alternate Recipient"
    / "Recipient MD Assigned Alternate Recipient"

```

It is derived from  
MTA.PerRecipientMessageTransferFields.extension.redirection-history.  
An example of this is:

```

X400-Recipients: postmaster@widget.com (Originally To:
    sales-manager@sales.widget.com Redirected
    on Thu, 30 May 91 14:39:40 +0100 To: Originator Assigned
    Alternate Recipient postmaster@sales.widget.com Redirected
    Again on Thu, 30 May 91 14:41:20 +0100 To: Recipient MD
    Assigned Alternate Recipient)

```

In addition, the following per-recipient services from MTS.OtherMessageDeliveryFields.extensions are represented in comments if they are used. None of these services can be provided on RFC 822 networks, and so in general these will be informative strings associated with other MTS recipients. In some cases, string values are defined. For the remainder, the string value shall be chosen by the implementor. If the parameter has a default value, then no comment shall be inserted when the parameter has that default value.

requested-delivery-method

physical-forwarding-prohibited  
"(Physical Forwarding Prohibited)".

physical-forwarding-address-request  
"(Physical Forwarding Address Requested)".

physical-delivery-modes

registered-mail-type

recipient-number-for-advice

physical-rendition-attributes

physical-delivery-report-request  
"(Physical Delivery Report Requested)".

proof-of-delivery-request  
"(Proof of Delivery Requested)".

#### 4.6.2.3. Delivery Report Generation

If MTA.PerRecipientMessageTransferFields.per-recipient-indicators requires a positive delivery notification, this shall be generated by the gateway. Supplementary Information shall be set to indicate that the report is gateway generated. This information shall include the name of the gateway generating the report.

#### 4.6.3. Message IDs (MTS)

A mapping from 822.msg-id to MTS.MTSIdentifier is defined. The reverse mapping is not needed, as MTS.MTSIdentifier is always mapped onto new RFC 822 fields. The value of MTS.MTSIdentifier.local-part will facilitate correlation of gateway errors.

To map from 822.msg-id, apply the standard mapping to 822.msg-id, in order to generate an MTS.ORAddress. The Country, ADMD, and PRMD components of this are used to generate MTS.MTSIdentifier.global-domain-identifier. MTS.MTSIdentifier.local-identifier is set to the 822.msg-id, including the braces "<" and ">". If this string is longer than MTS.ub-local-id-length (32), then it is truncated to this length.

The reverse mapping is not used in this specification. It would be applicable where MTS.MTSIdentifier.local-identifier is of syntax 822.msg-id, and it algorithmically identifies MTS.MTSIdentifier.

#### 4.7. IPMS Mappings

All RFC 822 addresses are assumed to use the 822.mailbox syntax. This includes all 822.comments associated with the lexical tokens of the 822.mailbox. In the IPMS O/R Names are encoded as MTS.ORName. This is used within the IPMS.ORDescriptor, IPMS.RecipientSpecifier, and IPMS.IPMIdentifier. An asymmetrical mapping is defined between these components.

##### 4.7.1. RFC 822 -> X.400

To derive IPMS.ORDescriptor from an RFC 822 address.

1. Take the address, and extract an EBNF.822-address. This can be derived trivially from either the 822.addr-spec or 822.route-addr syntax. This is mapped to MTS.ORName as described above, and used as IPMS.ORDescriptor.formal-name.
2. A string shall be built consisting of (if present):
  - The 822.phrase component if the 822.address is an 822.phrase 822.route-addr construct.
  - Any 822.comments, in order, retaining the parentheses.

This string is then encoded into T.61 use a human oriented mapping (as described in Chapter 3). If the string is not null, it is assigned to IPMS.ORDescriptor.free-form-name.

3. IPMS.ORDescriptor.telephone-number is omitted.

If IPMS.ORDescriptor is being used in IPMS.RecipientSpecifier, IPMS.RecipientSpecifier.reply-request and IPMS.RecipientSpecifier.notification-requests are set to default values (none and false).

If the 822.group construct is present, any included 822.mailbox is encoded as above to generate a separate IPMS.ORDescriptor. The 822.group is mapped to T.61, and a IPMS.ORDescriptor with only an free-form-name component built from it.

##### 4.7.2. X.400 -> RFC 822

Mapping from IPMS.ORDescriptor to RFC 822 address. In the basic case, where IPMS.ORDescriptor.formal-name is present, proceed as follows.

1. Encode IPMS.ORDescriptor.formal-name (MTS.ORName) as

EBNF.822-address.

- 2a. If IPMS.ORDescriptor.free-form-name is present, convert it to ASCII (Chapter 3), and use this as the 822.phrase component of 822.mailbox using the 822.phrase 822.route-addr construct.
- 2b. If IPMS.ORDescriptor.free-form-name is absent. If EBNF.822-address is parsed as 822.addr-spec use this as the encoding of 822.mailbox. If EBNF.822-address is parsed as 822.route 822.addr-spec, then a 822.phrase taken from 822.local-part is added.

3. If IPMS.ORDescriptor.telephone-number is present, this is placed in an 822.comment, with the string "Tel ". The normal international form of number is used. For example:

(Tel +44-1-387-7050)

4. If IPMS.ORDescriptor.formal-name.directory-name is present, then a text representation is placed in a trailing 822.comment.
5. If IPMS.RecipientSpecifier.report-request has any non-default values, then an 822.comment "(Receipt Notification Requested)", and/or "(Non Receipt Notification Requested)", and/or "(IPM Return Requested)" is appended to the address. If both receipt and non-receipt notifications are requested, the comment relating to the latter may be omitted, to make the RFC 822 address cleaner. The effort of correlating P1 and P2 information is too great to justify the gateway sending Receipt Notifications.
6. If IPMS.RecipientSpecifier.reply-request is True, an 822.comment "(Reply requested)" is appended to the address.

If IPMS.ORDescriptor.formal-name is absent, IPMS.ORDescriptor.free-form-name is converted to ASCII, and used as 822.phrase within the RFC 822 822.group syntax. For example:

Free Form Name ":" ";"

Steps 3-6 are then followed.

#### 4.7.3. IP Message IDs

There is a need to map both ways between 822.msg-id and IPMS.IPMIdentifier. This allows for X.400 Receipt Notifications,



Replies, and Cross References to reference an RFC 822 Message ID, which is preferable to a gateway generated ID. A reversible and symmetrical mapping is defined. This allows for good things to happen when messages pass multiple times across the X.400/RFC 822 boundary.

An important issue with messages identifiers is mapping to the exact form, as many systems use these ids as uninterpreted keys. The use of table driven mappings is not always symmetrical, particularly in the light of alternative domain names, and alternative management domains. For this reason, a purely algorithmic mapping is used. A mapping which is simpler than that for addresses can be used for two reasons:

- There is no major requirement to make message IDs "natural"
- There is no issue about being able to reply to message IDs. (For addresses, creating a return path which works is more important than being symmetrical).

The mapping works by defining a way in which message IDs generated on one side of the gateway can be represented on the other side in a systematic manner. The mapping is defined so that the possibility of clashes is low enough to be treated as impossible.

#### 4.7.3.1. 822.msg-id represented in X.400

IPMS.IPMIdentifier.user is omitted. The IPMS.IPMIdentifier.user-relative-identifier is set to a printable string encoding of the 822.msg-id with the angle braces ("<" and ">") removed. The upper bound on this component is 64. The options for handling this are discussed in Section 5.1.3.

#### 4.7.3.2. IPMS.IPMIdentifier represented in RFC 822

The 822.domain of 822.msg-id is set to the value "MHS". The 822.local-part of 822.msg-id is built as

```
[ printablestring ] "*" [ std-or-address ]
```

with EBNF.printablestring being the IPMS.IPMIdentifier.user-relative-identifier, and std-or-address being an encoding of the IPMS.IPMIdentifier.user. If necessary, the 822.quoted-string encoding is used. For example:

```
<"147*/S=Dietrich/O=Siemens/ADMD=DBP/C=DE/"@MHS>
```

#### 4.7.3.3. 822.msg-id -> IPMS.IPMIdentifier

If the 822.local-part can be parsed as:

```
[ printablestring ] "*" [ std-or-address ]
```

and the 822.domain is "MHS", then this ID was X.400 generated. If EBNF.printablestring is present, the value is assigned to IPMS.IPMIdentifier.user-relative-identifier. If EBNF.std-or-address is present, the O/R Address components derived from it are used to set IPMS.IPMIdentifier.user.

Otherwise, this is an RFC 822 generated ID. In this case, set IPMS.IPMIdentifier.user-relative-identifier to a printable string encoding of the 822.msg-id without the angle braces.

#### 4.7.3.4. IPMS.IPMIdentifier -> 822.msg-id

If IPMS.IPMIdentifier.user is absent, and IPMS.IPMIdentifier.user-relative-identifier mapped to ASCII and angle braces added parses as 822.msg-id, then this is an RFC 822 generated ID.

Otherwise, the ID is X.400 generated. Use the IPMS.IPMIdentifier.user to generate an EBNF.std-or-address form string. Build the 822.local-part of the 822.msg-id with the syntax:

```
[ printablestring ] "*" [ std-or-address ]
```

The printablestring is taken from IPMS.IPMIdentifier.user-relative-identifier. Use 822.quoted-string if necessary. The 822.msg-id is generated with this 822.local-part, and "MHS" as the 822.domain.

#### 4.7.3.5. Phrase form

In "InReply-To:" and "References:", the encoding 822.phrase may be used as an alternative to 822.msg-id. To map from 822.phrase to IPMS.IPMIdentifier, assign IPMS.IPMIdentifier.user-relative-identifier to the phrase. When mapping from IPMS.IPMIdentifier for "In-Reply-To:" and "References:", if IPMS.IPMIdentifier.user is absent and IPMS.IPMIdentifier.user-relative-identifier does not parse as 822.msg-id, generate an 822.phrase rather than adding the domain MHS.

#### 4.7.3.6. RFC 987 backwards compatibility

The mapping defined here is different to that used in RFC 987, as the RFC 987 mapping lead to changed message IDs in many cases. Fixing the problems is preferable to retaining backwards compatibility. An

implementation of this standard is encouraged to recognise message IDs generated by RFC 987. This is not required.

RFC 987 generated encodings may be recognised as follows. When mapping from X.400 to RFC 822, if the IPMS.IPMIdentifier.user-relative-identifier is "RFC-822" the id is RFC 987 generated. When mapping from RFC 822 to X.400, if the 822.domain is not "MHS", and the 822.local-part can be parsed as

```
[ printablestring ] "*" [ std-or-address ]
```

then it is RFC 987 generated. In each of these cases, it is recommended to follow the RFC 987 rules.

## Chapter 5 - Detailed Mappings

This chapter specifies detailed mappings for the functions outlined in Chapters 1 and 2. It makes extensive use of the notations and mappings defined in Chapters 3 and 4.

### 5.1. RFC 822 -> X.400

#### 5.1.1. Basic Approach

A single IP Message is generated from an RFC 822 message. The RFC 822 headers are used to generate the IPMS.Heading. The IP Message will have one IA5 IPMS.BodyPart containing the RFC 822 message body.

Some RFC 822 fields cannot be mapped onto a standard IPM Heading field, and so an extended field is defined in Section 5.1.2. This is then used for fields which cannot be mapped onto existing services.

The message is submitted to the MTS, and the services required can be defined by specifying MTS.MessageSubmissionEnvelope. A few parameters of the MTA Abstract service are also specified, which are not in principle available to the MTS User. Use of these services allows RFC 822 MTA level parameters to be carried in the analogous X.400 service elements. The advantages of this mapping far outweigh the layering violation.

#### 5.1.2. X.400 Extension Field

An IPMS Extension is defined:

```
rfc-822-field HEADING-EXTENSION
    VALUE RFC822FieldList
    ::= id-rfc-822-field-list
```

```
RFC822FieldList ::= SEQUENCE OF RFC822Field
```

```
RFC822Field ::= IA5String
```

The Object Identifier id-rfc-822-field-list is defined in Appendix D.

To encode any RFC 822 Header using this extension, an RFC822Field element is built using the 822.field omitting the trailing CRLF (e.g., "Fruit-Of-The-Day: Kiwi Fruit"). Structured fields shall be unfolded. There shall be no space before the ":". The reverse mapping builds the RFC 822 field in a straightforward manner. This RFC822Field is appended to the RFC822FieldList, which is added to the IPM Heading as an extension field.

### 5.1.3. Generating the IPM

The IPM (IPMS Service Request) is generated according to the rules of this section. The IPMS.IPM.body usually consists of one IPMS.BodyPart of type IPMS.IA5TextBodyPart with IPMS.IA5TextBodyPart.parameters.repertoire set to the default (ia5) which contains the body of the RFC 822 message. The exception is where there is a "Comments:" field in the RFC 822 header.

If no specific 1988 features are used, the IPM generated is encoded as content type 2. Otherwise, it is encoded as content type 22. The latter will always be the case if extension heading fields are generated.

When generating the IPM, the issue of upper bounds must be considered. At the MTS and MTA level, this specification is strict about enforcing upper bounds. Three options are available at the IPM level. Use of any of these options conforms to this standard.

1. Ignore upper bounds, and generate messages in the natural manner. This assumes that if any truncation is done, it will happen at the recipient UA. This will maximise transfer of information, but is likely break some recipient UAs.
2. Reject any inbound message which would cause a message violating constraints to be generated. This will be robust, but may prevent useful communication.
3. Truncate fields to the upper bounds specified in X.400.

This will prevent problems with UAs which enforce upper bounds, but will sometimes discard useful information.

If the Free Form name is truncated, it may lead to breaking RFC 822 comments, which will cause an awkward reverse mapping.

These options have different advantages and disadvantages, and the choice will depend on the exact application of the gateway.

The rest of this section concerns IPMS.IPM.heading (IPMS Heading). The only mandatory component of IPMS Heading is the IPMS Heading.this-IPM (IPMS.IPMIdentifier). A default is generated by the gateway. With the exception of "Received:", the values of multiple fields are merged (e.g., If there are two "To:" fields, then the mailboxes of both are merged to generate a single list which is used in the IPMS Heading.primary-recipients. Information shall be generated from the standard RFC 822 Headers as follows:

Date:

Ignore (Handled at MTS level)

Received:

Ignore (Handled at MTA level)

Message-Id:

Mapped to IPMS Heading.this-IPM. For these, and all other fields containing 822.msg-id the mappings of Chapter 4 are used for each 822.msg-id.

From:

If Sender: is present, this is mapped to IPMS Heading.authorizing-users. If not, it is mapped to IPMS Heading.originator. For this, and other components containing addresses, the mappings of Chapter 4 are used for each address.

Sender:

Mapped to IPMS Heading.originator.

Reply-To:

Mapped to IPMS Heading.reply-recipients.

To: Mapped to IPMS Heading.primary-recipients

Cc: Mapped to IPMS Heading.copy-recipients.

Bcc: Mapped to IPMS Heading.blind-copy-recipients if there is at least one BCC: recipient. If there are no recipients in this field, it should be mapped to a zero length sequence.

**In-Reply-To:**

If there is one value, it is mapped to `IPMS.Heading.replied-to-IPM`, using the `822.phrase` or `822.msg-id` mapping as appropriate. If there are several values, they are mapped to `IPMS.Heading.related-IPMs`, along with any values from a "References:" field.

**References:**

Mapped to `IPMS.Heading.related-IPMs`.

**Keywords:**

Mapped onto a heading extension.

**Subject:**

Mapped to `IPMS.Heading.subject`. The field-body uses the human oriented mapping referenced in Chapter 3 from ASCII to T.61.

**Comments:**

Generate an `IPMS.BodyPart` of type `IPMS.IA5TextBodyPart` with `IPMS.IA5TextBodyPart.parameters.repertoire` set to the default (`ia5`), containing the value of the fields, preceded by the string "Comments: ". This body part shall precede the other one.

**Encrypted:**

Mapped onto a heading extension.

**Resent-\***

Mapped onto a heading extension.

Note that it would be possible to use a `ForwardedIPMessage` for these fields, but the semantics are (arguably) slightly different, and it is probably not worth the effort.

**Other Fields**

In particular X-\* fields, and "illegal" fields in common usage (e.g., "Fruit-of-the-day:") are mapped onto a heading extension, unless covered by another section or appendix of this specification. The same treatment is applied to RFC 822 fields where the content of the field does not conform to RFC 822 (e.g., a `Date:` field with unparseable syntax).

**5.1.4. Mappings to the MTS Abstract Service**

The `MTS.MessageSubmissionEnvelope` comprises `MTS.PerMessageSubmissionFields`, and

MTS.PerRecipientMessageSubmissionFields. The mandatory parameters are defaulted as follows.

MTS.PerMessageSubmissionFields.originator-name

This is always generated from 822-MTS, as defined in Chapter 4.

MTS.PerMessageSubmissionFields.content-type

Set to the value implied by the encoding of the IPM (2 or 22).

MTS.PerRecipientMessageSubmissionFields.recipient-name

These will always be supplied from 822-MTS, as defined in Chapter 4.

Optional components are omitted, and default components defaulted. This means that disclosure of recipients is prohibited and conversion is allowed. There are two exceptions to the defaulting. For MTS.PerMessageSubmissionFields.per-message-indicators, the following settings are made:

- Alternate recipient is allowed, as it seems desirable to maximise the opportunity for (reliable) delivery.
- Content return request is set according to the issues discussed in Section 5.2.

MTS.PerMessageSubmissionFields.original-encoded-information-types is a set of one element BuiltInEncodedInformationTypes.ia5-text.

The MTS.PerMessageSubmissionFields.content-correlator is encoded as IA5String, and contains the Subject:, Message-ID:, Date:, and

To: fields (if present). This includes the strings "Subject:", "Date:", "To:", "Message-ID:", and appropriate folding. This shall be truncated to MTS.ub-content-correlator-length (512) characters. In addition, if there is a "Subject:" field, the MTS.PerMessageSubmissionFields.content-identifier, is set to a printable string representation of the contents of it. If the length of this string is greater than MTS.ub-content-id-length (16), it should be truncated to 13 characters and the string "... " appended. Both are used, due to the much larger upper bound of the content correlator, and that the content id is available in X.400(1984).

#### 5.1.5. Mappings to the MTA Abstract Service

There is a need to map directly onto some aspects of the MTA Abstract

service, for the following reasons:

- So the the MTS Message Identifier can be generated from the RFC 822 Message-ID:.
- So that the submission date can be generated from the 822.Date.
- To prevent loss of trace information
- To prevent RFC 822/X.400 looping caused by distribution lists or redirects

The following mappings are defined.

Message-Id:

If this is present, the  
MTA.PerMessageTransferFields.message-identifier is generated from it, using the mappings described in Chapter 4.

Date:

This is used to set the first component of  
MTA.PerMessageTransferFields.trace-information  
(MTA.TraceInformationElement). The 822-MTS originator is mapped into an MTS.ORAddress, and used to derive  
MTA.TraceInformationElement.global-domain-identifier. The optional components of  
MTA.TraceInformationElement.domain-supplied-information are omitted, and the mandatory components are set as follows:

MTA.DomainSuppliedInformation.arrival-time  
This is set to the date derived from Date:

MTA.DomainSuppliedInformation.routing-action  
Set to relayed.

The first element of  
MTA.PerMessageTransferFields.internal-trace-information is generated in an analogous manner, although this can be dropped later in certain circumstances (see the procedures for "Received:"). The  
MTA.InternalTraceInformationElement.mta-name is derived from the 822.domain in the 822 MTS Originator address.

Received:

All RFC 822 trace is used to derive  
MTA.PerMessageTransferFields.trace-information and  
MTA.PerMessageTransferFields.internal-trace-information.



Processing of Received: lines follows processing of Date:, and is be done from the the bottom to the top of the RFC 822 header (i.e., in chronological order). When other trace elements are processed (X400-Received: in all cases and Via: if Appendix B is supported), the relative ordering shall be retained correctly. The initial element of MTA.PerMessageTransferFields.trace-information will be generated already (from Date:), unless the message has previously been in X.400, when it will be derived from the X.400 trace information.

Consider the Received: field in question. If the "by" part of the received is present, use it to derive an MTS.GlobalDomainIdentifier. If this is different from the one in the last element of MTA.PerMessageTransferFields.trace-information (MTA.TraceInformationElement.global-domain-identifier) create a new MTA.TraceInformationElement, and optionally remove MTA.PerMessageTransferFields.internal-trace-information. This removal shall be done in cases where the message is being transferred to another MD where there is no bilateral agreement to preserve internal trace beyond the local MD. The trace creation is as for internal trace described below, except that no MTA field is needed.

Then add a new element (MTA.InternalTraceInformationElement) to MTA.PerMessageTransferFields.internal-trace-information, creating this if needed. This shall be done, even if inter-MD trace is created. The MTA.InternalTraceInformationElement.global-domain-identifier is set to the value derived. The MTA.InternalTraceInformationElement.mta-supplied-information (MTA.MTASuppliedInformation) is set as follows:

MTA.MTASuppliedInformation.arrival-time  
Derived from the date of the Received: line

MTA.MTASuppliedInformation.routing-action  
Set to relayed

The MTA.InternalTraceInformationElement.mta-name is taken from the "by" component of the "Received:" field, truncated to MTS.ub-mta-name-length (32). For example:

Received: from computer-science.nottingham.ac.uk by  
vs6.Cs.Ucl.AC.UK via Janet with NIFTP id aa03794;  
28 Mar 89 16:38 GMT

Generates the string

vs6.Cs.Ucl.AC.UK

Note that before transferring the message to some ADMDs, additional trace stripping may be required, as the implied path through multiple MDs would violate ADMD policy. This will depend on bilateral agreement with the ADMD.

#### 5.1.6. Mapping New Fields

This specification defines a number of new fields for Reports, Notifications and IP Messages in Section 5.3. As this specification only aims to preserve existing services, a gateway conforming to this specification does not need to map all of these fields to X.400.

Two extended fields must be mapped, in order to prevent looping. "DL-Expansion-History:" is mapped to

MTA.PerMessageTransferFields.extensions.dl-expansion-history X400-Received: must be mapped to MTA.PerMessageTransferFields.trace-information and MTA.PerMessageTransferFields.internal-trace-information. In cases where X400-Received: is present, the usual mapping of Date: to generate the first element of trace should not be done. This is because the message has come from X.400, and so the first element of trace can be taken from the first X400-Received:.

Some field that shall not be mapped, and should be discarded. The following cannot be mapped back:

- Discarded-X400-MTS-Extensions:
- Message-Type:
- Discarded-X400-IPMS-Extensions:

If Message-Type: is set to "Multiple Part", then the message is encoded according to RFC 934, and this may be mapped on to the corresponding X.400 structures.

The following may cause problems, due to other information not being mapped back (e.g., extension numbers), or due to changes made on the RFC 822 side due to list expansion:

- X400-Content-Type:
- X400-Originator:

- X400-Recipients:
- X400-MTS-Identifier:

Other fields may be either discarded or mapped to X.400. It is usually desirable and beneficial to do map, particularly to facilitate support of a message traversing multiple gateways. These mappings may be onto MTA, MTS, or IPMS services. The level of support for this reverse mapping should be indicated in the gateway conformance statement.

## 5.2. Return of Contents

It is not clear how widely supported the X.400 return of contents service will be. Experience with X.400(1984) suggests that support of this service may not be universal. As this service is expected in the RFC 822 world, two approaches are specified. The choice will depend on the use of X.400 return of contents within the X.400 community being serviced by the gateway.

In environments where return of contents is widely supported, content return can be requested as a service. The content return service can then be passed back to the end (RFC 822) user in a straightforward manner.

In environments where return of contents is not widely supported, a gateway must make special provision to handle return of contents. For every message passing from RFC 822 -> X.400, content return request will not be requested, and report request always will be. When the delivery report comes back, the gateway can note that the message has been delivered to the recipient(s) in question. If a non-delivery report is received, a meaningful report (containing some or all of the original message) can be sent to the 822-MTS originator. If no report is received for a recipient, a (timeout) failure notice shall be sent to the 822-MTS originator. The gateway may retransmit the X.400 message if it wishes. When this approach is taken, routing must be set up so that error reports are returned through the same MTA. This approach may be difficult to use in conjunction with some routing strategies.

## 5.3. X.400 -> RFC 822

### 5.3.1. Basic Approach

A single RFC 822 message is generated from the incoming IP Message, Report, or IP Notification. All IPMS.BodyParts are mapped onto a single RFC 822 body. Other services are mapped onto RFC 822 header fields. Where there is no appropriate existing field, new fields are

defined for IPMS, MTS and MTA services.

The gateway mechanisms will correspond to MTS Delivery. As with submission, there are aspects where the MTA (transfer) services are also used. In particular, there is an optimisation to allow for multiple 822-MTS recipients.

#### 5.3.2. RFC 822 Settings

An RFC 822 Service requires to have a number of mandatory fields in the RFC 822 Header. Some 822-MTS services mandate specification of an 822-MTS Originator. Even in cases where this is optional, it is usually desirable to specify a value. The following defaults are defined, which shall be used if the mappings specified do not derive a value:

##### 822-MTS Originator

If this is not generated by the mapping (e.g., for a Delivery Report), a value pointing at a gateway administrator shall be assigned.

##### Date:

A value will always be generated

From: If this is not generated by the mapping, it is assigned equal to the 822-MTS Originator. If this is gateway generated, an appropriate 822.phrase shall be added.

##### At least one recipient field

If no recipient fields are generated, a field "To: list;;", shall be added.

This will ensure minimal RFC 822 compliance. When generating RFC 822 headers, folding may be used. It is recommended to do this, following the guidelines of RFC 822.

#### 5.3.3. Basic Mappings

##### 5.3.3.1. Encoded Information Types

This mapping from MTS.EncodedInformationTypes is needed in several disconnected places. EBNF is defined as follows:

```
encoded-info      = 1#encoded-type
encoded-type      = built-in-eit / object-identifier
built-in-eit      = "Undefined"           ; undefined (0)
```

```

/ "Telex"                ; tLX (1)
/ "IA5-Text"             ; iA5Text (2)
/ "G3-Fax"               ; g3Fax (3)
/ "TIF0"                 ; tIF0 (4)
/ "Teletex"              ; tTX (5)
/ "Videotex"             ; videotex (6)
/ "Voice"                ; voice (7)
/ "SFD"                  ; sFD (8)
/ "TIF1"                 ; tIF1 (9)

```

MTS.EncodedInformationTypes is mapped onto EBNF.encoded-info.  
 MTS.EncodedInformationTypes.non-basic-parameters is ignored. Built  
 in types are mapped onto fixed strings (compatible with X.400(1984)  
 and RFC 987), and other types are mapped onto EBNF.object-identifier.

#### 5.3.3.2. Global Domain Identifier

The following simple EBNF is used to represent  
 MTS.GlobalDomainIdentifier:

```
global-id = std-or-address
```

This is encoded using the std-or-address syntax, for the attributes  
 within the Global Domain Identifier.

#### 5.3.4. Mappings from the IP Message

Consider that an IPM has to be mapped to RFC 822. The IPMS.IPM  
 comprises an IPMS.IPM.heading and IPMS.IPM.body. The heading is  
 considered first. Some EBNF for new fields is defined:

```

ipms-field = "Obsoletes" ":" 1#msg-id
/ "Expiry-Date" ":" date-time
/ "Reply-By" ":" date-time
/ "Importance" ":" importance
/ "Sensitivity" ":" sensitivity
/ "Autoforwarded" ":" boolean
/ "Incomplete-Copy" ":"
/ "Language" ":" language
/ "Message-Type" ":" message-type
/ "Discarded-X400-IPMS-Extensions" ":" 1#oid

```

```
importance      = "low" / "normal" / "high"
```

```
sensitivity     = "Personal" / "Private" /
```

"Company-Confidential"

language = 2\*ALPHA [ language-description ]  
language-description = printable-string

message-type = "Delivery Report"  
/ "InterPersonal Notification"  
/ "Multiple Part"

The mappings and actions for the IPMS.Heading is now specified for each element. Addresses, and Message Identifiers are mapped according to Chapter 4. Other mappings are explained, or are straightforward (algorithmic). If a field with addresses contains zero elements, it should be discarded, except for IPMS.Heading.blind-copy-recipients, which can be mapped onto BCC: (the only RFC 822 field which allows zero recipients).

IPMS.Heading.this-IPM  
Mapped to "Message-ID:".

IPMS.Heading.originator  
If IPMS.Heading.authorizing-users is present this is mapped to Sender:, if not to "From:".

IPMS.Heading.authorizing-users  
Mapped to "From:".

IPMS.Heading.primary-recipients  
Mapped to "To:".

IPMS.Heading.copy-recipients  
Mapped to "Cc:".

IPMS.Heading.blind-copy-recipients  
Mapped to "Bcc:".

IPMS.Heading.replied-to-ipm  
Mapped to "In-Reply-To:".

IPMS.Heading.obsoleted-IPMs  
Mapped to the extended RFC 822 field "Obsoletes:"

IPMS.Heading.related-IPMs  
Mapped to "References:".

IPMS.Heading.subject

Mapped to "Subject:". The contents are converted to ASCII (as defined in Chapter 3). Any CRLF are not mapped, but are used as points at which the subject field must be folded.

IPMS.Heading.expiry-time

Mapped to the extended RFC 822 field "Expiry-Date:".

IPMS.Heading.reply-time

Mapped to the extended RFC 822 field "Reply-By:".

IPMS.Heading.reply-recipients

Mapped to "Reply-To:".

IPMS.Heading.importance

Mapped to the extended RFC 822 field "Importance:".

IPMS.Heading.sensitivity

Mapped to the extended RFC 822 field "Sensitivity:".

IPMS.Heading.autoforwarded

Mapped to the extended RFC 822 field "Autoforwarded:".

The standard extensions (Annex H of X.420 / ISO 10021-7) are mapped as follows:

incomplete-copy

Mapped to the extended RFC 822 field "Incomplete-Copy:".

language

Mapped to the extended RFC 822 field "Language:", filling in the two letter code. The language-description may filled in with a human readable description of the language, and it is recommended to do this.

If the RFC 822 extended header is found, this shall be mapped onto an RFC 822 header, as described in Section 5.1.2.

If a non-standard extension is found, it shall be discarded, unless the gateway understands the extension and can perform an appropriate mapping onto an RFC 822 header field. If extensions are discarded, the list is indicated in the extended RFC 822 field "Discarded-X400-IPMS-Extensions:".

The IPMS.Body is mapped into the RFC 822 message body. Each IPMS.BodyPart is converted to ASCII as follows:

**IPMS.IA5Text**

The mapping is straightforward (see Chapter 3).

**IPMS.MessageBodyPart**

The X.400 -> RFC 822 mapping is recursively applied, to generate an RFC 822 Message. If present, the `IPMS.MessageBodyPart.parameters.delivery-envelope` is used for the MTS Abstract Service Mappings. If present, the `IPMS.MessageBodyPart.parameters.delivery-time` is mapped to the extended RFC 822 field "Delivery-Date:".

**Other**

If other body parts can be mapped to IA5, either by use of mappings defined in X.408 [CCITT88a], or by other reasonable mappings, this shall be done unless content conversion is prohibited.

If some or all of the body parts cannot be converted there are three options. All of these conform to this standard. A different choice may be made for the case where no body part can be converted:

1. The first option is to reject the message, and send a non-delivery notification. This must always be done if conversion is prohibited.
2. The second option is to map a missing body part to something of the style:

\*\*\*\*\*

There was a foobarhere

The widget gateway ate it

\*\*\*\*\*

This will allow some useful information to be transferred. As the recipient is likely to be a human (IPMS), then suitable action will usually be possible.

3. Finally both may be done. In this case, the supplementary information in the (positive) Delivery Report shall make clear that something was sent on to the recipient with substantial loss of information.

Where there is more than one `IPMS.BodyPart`, the mapping defined by Rose and Stefferud in [Rose85a], is used to map the separate `IPMS.BodyParts` in the single RFC 822 message body. If this is done,



a "Message-Type:" field with value "Multiple part" shall be added, which will indicate to a receiving gateway that the message may be unfolded according to RFC 934.

Note: There is currently work ongoing to produce an upgrade to RFC 934, which also allows for support of body parts with non-ASCII content (MIME). When this work is released as an RFC, this specification will be updated to refer to it instead for RFC 934.

For backwards compatibility with RFC 987, the following procedures shall also be followed. If there are two IA5 body parts, and the first starts with the string "RFC-822-Headers:" as the first line, then the remainder of this body part shall be appended to the RFC 822 header.

An example message, illustrating a number of aspects is given below.

```
Return-Path:<@mhs-relay.ac.uk:stephen.harrison@gossip-uk.hmg.gold-400.gb>
Received: from mhs-relay.ac.uk by bells.cs.ucl.ac.uk via JANET
        with NIFTP id <7906-0@bells.cs.ucl.ac.uk>;
        Thu, 30 May 1991 18:24:55 +0100
X400-Received: by mta "mhs-relay.ac.uk" in
        /PRMD=uk.ac/ADMD= /C=gb/; Relayed;
        Thu, 30 May 1991 18:23:26 +0100
X400-Received: by /PRMD=HMG/ADMD=GOLD 400/C=GB/; Relayed;
        Thu, 30 May 1991 18:20:27 +0100
Message-Type: Multiple Part
Date: Thu, 30 May 1991 18:20:27 +0100
X400-Originator: Stephen.Harrison@gossip-uk.hmg.gold-400.gb
X400-MTS-Identifier:
        [/PRMD=HMG/ADMD=GOLD 400/C=GB/;PC1000-910530172027-57D8]
Original-Encoded-Information-Types: ia5, undefined
X400-Content-Type: P2-1984 (2)
Content-Identifier: Email Problems
From: Stephen.Harrison@gossip-uk.hmg.gold-400.gb (Tel +44 71 217 3487)
Message-ID: <PC1000-910530172027-57D8*@MHS>
To: Jim Craigie <NTIN36@gec-b.rutherford.ac.uk>
    (Receipt Notification Requested) (Non Receipt Notification Requested),
    Tony Bates <tony@ean-relay.ac.uk> (Receipt Notification Requested),
    Steve Kille <S.Kille@cs.ucl.ac.uk> (Receipt Notification Requested)
Subject: Email Problems
Sender: Stephen.Harrison@gossip-uk.hmg.gold-400.gb
```

----- Start of body part 1

Hope you gentlemen.....

Regards,

Stephen Harrison  
UK GOSIP Project

----- Start of forwarded message 1

From: Urs Eppenberger <Eppenberger@verw.switch.ch>

Message-ID:

<562\*/S=Eppenberger/OU=verw/O=switch/PRMD=SWITCH/ADMD=ARCOM/C=CH/@MHS>

To: "Stephen.Harrison" <Stephen.Harrison@gossip-uk.hmg.gold-400.gb>

Cc: kimura@bsdarc.bsd.fc.nec.co.jp

Subject: Response to Email link

- ----- Start of body part 1

Dear Mr Harrison.....

- ----- End of body part 1

----- End of forwarded message 1

#### 5.3.5. Mappings from an IP Notification

A message is generated, with the following fields:

From:

Set to the IPMS.IPN.ipn-originator.

To:

Set to the recipient from MTS.MessageSubmissionEnvelope.  
If there have been redirects, the original address should be used.

Subject:

Set to the string "X.400 Inter-Personal Notification" for a receipt notification and to "X.400 Inter-Personal Notification (failure)" for a non-receipt notification.

Message-Type:

Set to "InterPersonal Notification"

References:

Set to IPMS.IPN.subject-ipm

The following EBNF is defined for the body of the Message. This format is defined to ensure that all information from an

interpersonal notification is available to the end user in a uniform manner.

```

ipn-body-format = ipn-description <CRLF>
                  [ ipn-extra-information <CRLF> ]
                  [ ipn-content-return ]

ipn-description = ipn-receipt / ipn-non-receipt

ipn-receipt = "Your message to:" preferred-recipient <CRLF>
              "was received at" receipt-time <CRLF> <CRLF>
              "This notification was generated"
              acknowledgement-mode <CRLF>
              "The following extra information was given:" <CRLF>
              ipn-suppl <CRLF>

ipn-non-receipt "Your message to:"
                preferred-recipient <CRLF>
                ipn-reason

ipn-reason = ipn-discarded / ipn-auto-forwarded

ipn-discarded = "was discarded for the following reason:"
                discard-reason <CRLF>

ipn-auto-forwarded = "was automatically forwarded." <CRLF>
                    [ "The following comment was made:"
                      auto-comment ]

ipn-extra-information =
    "The following information types were converted:"
    encoded-info

ipn-content-return = "The Original Message is not available"
                    / "The Original Message follows:"
                    <CRLF> <CRLF> message

preferred-recipient = mailbox
receipt-time        = date-time
auto-comment        = printablestring
ipn-suppl           = printablestring

discard-reason      = "Expired" / "Obsoleted" /
                    "User Subscription Terminated"

```

acknowledgement-mode = "Manually" / "Automatically"

The mappings for elements of the common fields of IPMS.IPN (IPMS.CommonFields) onto this structure and the message header are:

subject-ipm

Mapped to "References:"

ipn-originator

Mapped to "From:".

ipn-preferred-recipient

Mapped to EBNF.preferred-recipient

conversion-eits

Mapped to EBNF.encoded-info in EBNF.ipn-extra-information

The mappings for elements of IPMS.IPN.non-receipt-fields (IPMS.NonReceiptFields) are:

non-receipt-reason

Used to select between EBNF.ipn-discarded and EBNF.ipn-auto-forwarded

discard-reason

Mapped to EBNF.discard-reason

auto-forward-comment

Mapped to EBNF.auto-comment

returned-ipm

This applies only to non-receipt notifications. EBNF.ipn-content-return should always be omitted for receipt notifications, and always be present in non-receipt notifications. If present, the second option of EBNF.ipn-content-return is chosen, and an RFC 822 mapping of the message included. Otherwise the first option is chosen.

The mappings for elements of IPMS.IPN.receipt-fields (IPMS.ReceiptFields) are:

receipt-time

Mapped to EBNF.receipt-time

acknowledgement-mode

Mapped to EBNF.acknowledgement-mode

suppl-receipt-info  
Mapped to EBNF.ipn-suppl

An example notification is:

```
From: Steve Kille <steve@cs.ucl.ac.uk>
To: Julian Onions <jpo@computer-science.nottingham.ac.uk>
Subject: X.400 Inter-personal Notification
Message-Type: InterPersonal Notification
References: <1229.614418325@UK.AC.NOTT.CS>
Date: Wed, 21 Jun 89 08:45:25 +0100
```

```
Your message to: Steve Kille <steve@cs.ucl.ac.uk>
was automatically forwarded.
The following comment was made:
    Sent on to a random destination
```

The following information types were converted: g3fax

### 5.3.6. Mappings from the MTS Abstract Service

This section describes the MTS mappings for User Messages (IPM and IPN). This mapping is defined by specifying the mapping of MTS.MessageDeliveryEnvelope. The following extensions to RFC 822 are defined to support this mapping:

```
mts-field = "X400-MTS-Identifier" ":" mts-msg-id
           / "X400-Originator" ":" mailbox
           / "X400-Recipients" ":" 1#mailbox
           / "Original-Encoded-Information-Types" ":"
             encoded-info
           / "X400-Content-Type" ":" mts-content-type
           / "Content-Identifier" ":" printablestring
           / "Priority" ":" priority
           / "Originator-Return-Address" ":" 1#mailbox
           / "DL-Expansion-History" ":" mailbox ";" date-time ";"
           / "Conversion" ":" prohibition
           / "Conversion-With-Loss" ":" prohibition
           / "Requested-Delivery-Method" ":"
             1*( labelled-integer )
           / "Delivery-Date" ":" date-time
           / "Discarded-X400-MTS-Extensions" ":"
             1#( oid / labelled-integer )

prohibition      = "Prohibited" / "Allowed"

mts-msg-id       = "[" global-id ";" *text "]"
```

```
mts-content-type = "P2" / labelled-integer  
                  / object-identifier
```

```
priority          = "normal" / "non-urgent" / "urgent"
```

The mappings for each element of MTS.MessageDeliveryEnvelope can now be considered.

```
MTS.MessageDeliveryEnvelope.message-delivery-identifier  
  Mapped to the extended RFC 822 field "X400-MTS-Identifier:".
```

```
MTS.MessageDeliveryEnvelope.message-delivery-time  
  Discarded, as this time will be represented in an  
  appropriate trace element.
```

The mappings for elements of  
MTS.MessageDeliveryEnvelope.other-fields  
(MTS.OtherMessageDeliveryFields) are:

```
content-type  
  Mapped to the extended RFC 822 field "X400-Content-Type:".  
  The string "P2" is retained for backwards compatibility with  
  RFC 987. This shall not be generated, and either the  
  EBNF.labelled-integer or EBNF.object-identifier encoding  
  used.
```

```
originator-name  
  Mapped to the 822-MTS originator, and to the extended RFC  
  822 field "X400-Originator:". This is described in  
  Section 4.6.2.
```

```
original-encoded-information-types  
  Mapped to the extended RFC 822 field  
  "Original-Encoded-Information-Types:".
```

```
priority  
  Mapped to the extended RFC 822 field "Priority:".
```

```
delivery-flags  
  If the conversion-prohibited bit is set, add an extended RFC  
  822 field "Conversion:".
```

this-recipient-name and other-recipient-names

```
originally-intended-recipient-name  
  The handling of these elements is described in  
  Section 4.6.2.
```

converted-encoded-information-types

Discarded, as it will always be IA5 only.

message-submission-time

Mapped to Date:.

content-identifier

Mapped to the extended RFC 822 field "Content-Identifier:".

If any extensions (MTS.MessageDeliveryEnvelope.other-fields.extensions) are present, and they are marked as critical for transfer or delivery, then the message shall be rejected. The extensions (MTS.MessageDeliveryEnvelope.other-fields.extensions) are mapped as follows.

conversion-with-loss-prohibited

If set to

MTS.ConversionWithLossProhibited.conversion-with-loss-prohibited, then add the extended RFC 822 field "Conversion-With-Loss:".

requested-delivery-method

Mapped to the extended RFC 822 field  
"Requested-Delivery-Method:".

originator-return-address

Mapped to the extended RFC 822 field  
"Originator-Return-Address:".

physical-forwarding-address-request

physical-delivery-modes

registered-mail-type

recipient-number-for-advice

physical-rendition-attributes

physical-delivery-report-request

physical-forwarding-prohibited

These elements are only appropriate for physical delivery. They are represented as comments in the "X400-Recipients:" field, as described in Section 4.6.2.2.

originator-certificate

message-token

content-confidentiality-algorithm-identifier

content-integrity-check

message-origin-authentication-check

message-security-label

proof-of-delivery-request

These elements imply use of security services not available in the RFC 822 environment. If they are marked as critical for transfer or delivery, then the message shall be rejected. Otherwise they are discarded.

#### redirection-history

This is described in Section 4.6.2.

#### dl-expansion-history

Each element is mapped to the extended RFC 822 field "DL-Expansion-History:". They shall be ordered in the message header, so that the most recent expansion comes first (same order as trace).

If any MTS (or MTA) Extensions not specified in X.400 are present, and they are marked as critical for transfer or delivery, then the message shall be rejected. If they are not so marked, they can safely be discarded. The list of discarded fields shall be indicated in the extended header "Discarded-X400-MTS-Extensions:".

### 5.3.7. Mappings from the MTA Abstract Service

There are some mappings at the MTA Abstract Service level which are done for IPM and IPN. These can be derived from MTA.MessageTransferEnvelope. The reasons for the mappings at this level, and the violation of layering are:

- Allowing for multiple recipients to share a single RFC 822 message
- Making the X.400 trace information available on the RFC 822 side
- Making any information on deferred delivery available

The 822-MTS recipients are calculated from the full list of X.400 recipients. This is all of the members of MTA.MessageTransferEnvelope.per-recipient-fields being passed through the gateway, where the responsibility bit is set. In some cases, a different RFC 822 message would be calculated for each recipient, due to differing service requests for each recipient. As discussed in 4.6.2..2, this specification allows either for multiple messages to be generated, or for the per-recipient information to be discarded.

The following EBNF is defined for extended RFC 822 headers:

```
mta-field      = "X400-Received" ":" x400-trace
               / "Deferred-Delivery" ":" date-time
```



```

/ "Latest-Delivery-Time" ":" date-time

x400-trace      = "by" md-and-mta ";"
                  [ "deferred until" date-time ";" ]
                  [ "converted" "(" encoded-info ")" ";" ]
                  [ "attempted" md-or-mta ";" ]
                  action-list
                  ";" arrival-time

md-and-mta      = [ "mta" mta "in" ] global-id
mta              = word
arrival-time     = date-time

md-or-mta       = "MD" global-id
                  / "MTA" mta

Action-list     = 1#action
action          = "Redirected"
                  / "Expanded"
                  / "Relayed"
                  / "Rerouted"

```

Note the EBNF.mta is encoded as 822.word. If the character set does not allow encoding as 822.atom, the 822.quoted-string encoding is used.

If MTA.PerMessageTransferFields.deferred-delivery-time is present, it is used to generate a Deferred-Delivery: field. For some reason, X.400 does not make this information available at the MTS level on delivery. X.400 profiles, and in particular the CEN/CENELEC profile for X.400(1984) [Systems85a], specify that this element must be supported at the first MTA. If it is not, the function may optionally be implemented by the gateway: that is, the gateway may hold the message until the time specified in the protocol element. Thus, the value of this element will usually be in the past. For this reason, the extended RFC 822 field is primarily for information.

Merge MTA.PerMessageTransferFields.trace-information, and MTA.PerMessageTransferFields.internal-trace-information to produce a single ordered trace list. If Internal trace from other management domains has not been stripped, this may require complex interleaving. Where an element of internal trace and external trace are identical, except for the MTA in the internal trace, only the internal trace element shall be presented. Use this to generate a sequence of "X400-Received:" fields. The only difference between external trace and internal trace will be the extra MTA information in internal trace elements.

When generating an RFC 822 message all trace fields (X400-Received and Received) shall be at the beginning of the header, before any other fields. Trace shall be in chronological order, with the most recent element at the front of the message. This ordering is determined from the order of the fields, not from timestamps in the trace, as there is no guarantee of clock synchronisation. A simple example trace (external) is:

```
X400-Received: by /PRMD=UK.AC/ADMD=Gold 400/C=GB/ ; Relayed ;  
    Tue, 20 Jun 89 19:25:11 +0100
```

A more complex example (internal):

```
X400-Received: by mta "UK.AC.UCL.CS"  
    in /PRMD=UK.AC/ADMD=Gold 400/C=GB/ ;  
    deferred until Tue, 20 Jun 89 14:24:22 +0100 ;  
    converted (undefined, g3fax) ";" attempted /ADMD=Foo/C=GB/ ;  
    Relayed, Expanded, Redirected ; Tue, 20 Jun 89 19:25:11 +0100
```

#### 5.3.8. Mappings from Report Delivery

Delivery reports are mapped at the MTS service level. This means that only reports destined for the MTS user will be mapped. Some additional services are also taken from the MTA service.

##### 5.3.8.1. MTS Mappings

A Delivery Report service will be represented as MTS.ReportDeliveryEnvelope, which comprises of per-report-fields (MTS.PerReportDeliveryFields) and per-recipient-fields.

A message is generated with the following fields:

**From:**

An administrator at the gateway system. This is also the 822-MTS originator.

**To:**

A mapping of the MTA.ReportTransferEnvelope.report-destination-name. This is also the 822-MTS recipient.

**Message-Type:**

Set to "Delivery Report".

**Subject:**

The EBNF for the subject line is:

```

subject-line  = "Delivery-Report" "(" status ")"
               [ "for" destination ]

status        = "success" / "failure" / "success and failures"

destination   = mailbox / "MTA" word

```

The format of the body of the message is defined to ensure that all information is conveyed to the RFC 822 user in a consistent manner. The format is structured as if it was a message coming from X.400, with the description in one body part, and a forwarded message (return of content) in the second. This structure is useful to the RFC 822 recipient, as it enables the original message to be extracted. The first body part is structured as follows:

1. A few lines giving keywords to indicate the original message.
2. A human summary of the status of each recipient being reported on.
3. A clearly marked section which contains detailed information extracted from the report. This is marked clearly, as it will not be comprehensible to the average user. It is retained, as it may be critical to diagnosing an obscure problem.

This section may be omitted in positive DRs, and it is recommended that this is appropriate for most gateways.

```

dr-body-format = dr-summary <CRLF>
                 dr-recipients <CRLF>
                 dr-administrator-info-envelope <CRLF>
                 dr-content-return

```

```

dr-content-return = "The Original Message is not available"
                   / "The Original Message follows:"

```

```

dr-summary = "This report relates to your message:" <CRLF>
             content-correlator <CRLF> <CRLF>
             "of" date-time <CRLF> <CRLF>

```

```

dr-recipients = *(dr-recipient <CRLF> <CRLF>)

```

```

dr-recipient = dr-recipient-success / dr-recipient-failure

```

```

dr-recv-success =
    "Your message was successfully delivered to:"
    mailbox "at" date-time

dr-recv-failure = "Your message was not delivered to:"
    mailbox <CRLF>
    "for the following reason:" *word

dr-administrator-info-envelope = 3*( "*" text <CRLF> )

dr-administrator-info =
    "***** The following information is directed towards"
    "the local administrator" <CRLF>
    "***** and is not intended for the end user" <CRLF> <CRLF>
    "DR generated by:" report-point <CRLF>
    "at" date-time <CRLF> <CRLF>
    "Converted to RFC 822 at" mta <CRLF>
    "at" date-time <CRLF> <CRLF>
    "Delivery Report Contents:" <CRLF> <CRLF>
    drc-field-list <CRLF>
    "***** End of administration information"

drc-field-list      = *(drc-field <CRLF>)

drc-field = "Subject-Submission-Identifier" ":"
    mts-msg-id
    / "Content-Identifier" ":" printablestring
    / "Content-Type" ":" mts-content-type
    / "Original-Encoded-Information-Types" ":"
    encoded-info
    / "Originator-and-DL-Expansion-History" ":"
    dl-history
    / "Reporting-DL-Name" ":" mailbox
    / "Content-Correlator" ":" content-correlator
    / "Recipient-Info" ":" recipient-info
    / "Subject-Intermediate-Trace-Information" ":"
    x400-trace

recipient-info = mailbox "," std-or ";"
    report-type
    [ "converted eits" encoded-info ";" ]
    [ "originally intended recipient"
    mailbox "," std-or ";" ]
    [ "last trace" [ encoded-info ] date-time ";" ]

```

```
[ "supplementary info" <"> printablestring <"> ";" ]
[ "redirection history" 1#redirection ";" ]
[ "physical forwarding address"
    printablestring ";" ]
```

```
report-type      = "SUCCESS" drc-success
                  / "FAILURE" drc-failure

drc-success      = "delivered at" date-time ";"
                  [ "type of MTS user" labelled-integer ";" ]

drc-failure      = "reason" labelled-integer ";"
                  [ "diagnostic" labelled-integer ";" ]

report-point = [ "mta" word "in" ] global-id
content-correlator = *word
dl-history = 1#( mailbox "(" date-time ")" )
```

The format is defined as a fixed definition of an the outer level (EBNF.dr-body-format). The element EBNF.dr-administrator-info-envelope, provides a means of encapsulating a section of the header in a manner which is clear to the end user. Each line of this section begins with "\*". Each element of EBNF.text within %EBNF.dr-administrator-info-envelope must not contain <CRLF>. This is used to wrap up EBNF.dr-administrator-info, which will generate a sequence of lines not starting with "\*". EBNF.drc-fields may be folded using the RFC 822 folding rules.

The elements of MTS.ReportDeliveryEnvelope.per-report-fields are mapped as follows onto extended RFC 822 fields:

```
subject-submission-identifier
    Mapped to EBNF.drc-field (Subject-Submission-Identifier)

content-identifier
    Mapped to EBNF.drc-field (Content-Identifier). This should
    also be used in EBNF.dr-summary if there is no Content
    Correlator present.

content-type
    Mapped to EBNF.drc-field (Content-Type)

original-encoded-information-types
    Mapped to EBNF.drc-field (Encoded-Info)
```

The extensions from MTS.ReportDeliveryEnvelope.per-report-fields.extensions are mapped as follows:

originator-and-DL-expansion-history

Mapped to EBNF.drc-field (Originator-and-DL-Expansion-History)

reporting-DL-name

Mapped to EBNF.drc-field (Reporting-DL-Name)

content-correlator

Mapped to EBNF.content-correlator, provided that the encoding is IA5String (this will always be the case). This is used in EBNF.dr-summary and EBNF.drc-field-list. In the former, LWSP may be added, in order to improve the layout of the message.

message-security-label reporting-MTA-certificate report-origin-authentication-check

These security parameters will not be present unless there is an error in a remote MTA. If they are present, they shall be discarded in preference to discarding the whole report.

For each element of MTS.ReportDeliveryEnvelope.per-recipient-fields, a value of EBNF.dr-recipient, and an EBNF.drc-field (Recipient-Info) is generated. The components are mapped as follows.

actual-recipient-name

Used to generate the first EBNF.mailbox and EBNF.std-or in EBNF.recipient-info. Both RFC 822 and X.400 forms are given, as there may be a problem in the mapping tables. It also generates the EBNF.mailbox in EBNF.dr-recipient-success or EBNF.dr-recipient-failure.

report

If it is MTS.Report.delivery, then set EBNF.dr-recipient to EBNF.dr-recipient-success, and similarly set EBNF.report-type, filling in EBNF.drc-success. If it is a failure, set EBNF.dr-recipient to EBNF.dr-recipient-failure, making a human interpretation of the reason and diagnostic codes, and including any supplementary information. EBNF.drc-failure is filled in systematically.

converted-encoded-information-types

Set EBNF.drc-field ("converted eits")

originally-intended-recipient

Set the second ("originally intended recipient") mailbox and  
std-or in EBNF.drc-field.

supplementary-info

Set EBNF.drc-field ("supplementary info"), and include this  
information in EBNF.dr-recv-failure.

redirection-history

Set EBNF.drc-field ("redirection history")

physical-forwarding-address

Set EBNF.drc-field ("physical forwarding address")

recipient-certificate

Discard

proof-of-delivery

Discard

Any unknown extensions shall be discarded, irrespective of  
criticality.

The original message, or an extract from it, shall be included in the  
delivery port if it is available. The original message will usually  
be available at the gateway, as discussed in Section 5.2. If the  
original message is available, but of erroneous format, a dump of the  
ASN.1 may be included. This is recommended, but not required.

#### 5.3.8.2. MTA Mappings

The single 822-MTS recipient is constructed from  
MTA.ReportTransferEnvelope.report-destination-name, using the  
mappings of Chapter 4. Unlike with a user message, this information  
is not available at the MTS level.

The following additional mappings are made:

MTA.ReportTransferEnvelope.report-destination-name

This is used to generate the To: field.

MTA.ReportTransferEnvelope.identifier

Mapped to the extended RFC 822 field "X400-MTS-Identifier:".

It may also be used to derive a "Message-Id:" field.

MTA.ReportTransferEnvelope.trace-information

and

MTA.ReportTransferEnvelope.internal-trace-information  
 Mapped onto the extended RFC 822 field "X400-Received:", as  
 described in Section 5.3.7. The first element is also used  
 to generate the "Date:" field, and the EBNF.report-point.

MTA.PerRecipientReportTransferFields.last-trace-information  
 Mapped to EBNF.recipient-info (last trace)

MTA.PerReportTransferFields.subject-intermediate-trace-  
 information Mapped to EBNF.drc-field (Subject-Intermediate-  
 Trace-Information). These fields are ordered so that the  
 most recent trace element comes first.

### 5.3.8.3. Example Delivery Reports

Example Delivery Report 1:

```
Return-Path: <postmaster@cs.ucl.ac.uk>
Received: from cs.ucl.ac.uk by bells.cs.ucl.ac.uk
    via Delivery Reports Channel id <27699-0@bells.cs.ucl.ac.uk>;
    Thu, 7 Feb 1991 15:48:39 +0000
From: UCL-CS MTA <postmaster@cs.ucl.ac.uk>
To: S.Kille@cs.ucl.ac.uk
Subject: Delivery Report (failure) for H.Hildegard@bbn.com
Message-Type: Delivery Report
Date: Thu, 7 Feb 1991 15:48:39 +0000
Message-ID: <"bells.cs.u.694:07.01.91.15.48.34"@cs.ucl.ac.uk>
Content-Idenfifier: Greetings.
```

----- Start of body part 1

This report relates to your message: Greetings.  
 of Thu, 7 Feb 1991 15:48:20 +0000

Your message was not delivered to  
 H.Hildegard@bbn.com for the following reason:  
 Bad Address  
 MTA 'bbn.com' gives error message (USER) Unknown user  
 name in "H.Hildegard@bbn.com"

```
***** The following information is directed towards the local
***** administrator and is not intended for the end user
*
* DR generated by mta bells.cs.ucl.ac.uk
*      in /PRMD=uk.ac/ADMD=gold 400/C=gb/
*      at Thu, 7 Feb 1991 15:48:34 +0000
```



\*  
\* Converted to RFC 822 at bells.cs.ucl.ac.uk  
\* at Thu, 7 Feb 1991 15:48:40 +0000  
\*  
\* ..... continued on next page  
  
\* Delivery Report Contents:  
\*  
\* Subject-Submission-Identifier:  
\* [/PRMD=uk.ac/ADMD=gold 400/C=gb/;<1803.665941698@UK.AC.UCL.CS>]  
\* Content-Identifier: Greetings.  
\* Subject-Intermediate-Trace-Information:  
\* /PRMD=uk.ac/ADMD=gold 400/C=gb/;  
\* arrival Thu, 7 Feb 1991 15:48:20 +0000 action Relayed  
  
\* Subject-Intermediate-Trace-Information:  
\* /PRMD=uk.ac/ADMD=gold 400/C=gb/;  
\* arrival Thu, 7 Feb 1991 15:48:18 +0000 action Relayed  
\* Recipient-Info: H.Hildegard@bbn.com,  
\* /RFC-822=H.Hildegard(a)bbn.com/OU=cs/O=ucl  
\* /PRMD=uk.ac/ADMD=gold 400/C=gb/;  
\* FAILURE reason Unable-To-Transfer (1);  
\* diagnostic Unrecognised-ORName (0);  
\* last trace (ia5) Thu, 7 Feb 1991 15:48:18 +0000;  
\* supplementary info "MTA 'bbn.com' gives error message (USER)  
\* Unknown user name in "H.Hildegard@bbn.com"";  
\*\*\*\*\* End of administration information

The Original Message follows:

----- Start of forwarded message 1

Received: from glenlivet.cs.ucl.ac.uk by bells.cs.ucl.ac.uk  
with SMTP inbound id <27689-0@bells.cs.ucl.ac.uk>;  
Thu, 7 Feb 1991 15:48:21 +0000  
To: H.Hildegard@bbn.com  
Subject: Greetings.  
Phone: +44-71-380-7294  
Date: Thu, 07 Feb 91 15:48:18 +0000  
Message-ID: <1803.665941698@UK.AC.UCL.CS>  
From: Steve Kille <S.Kille@cs.ucl.ac.uk>

Steve

----- End of forwarded message 1  
Example Delivery Report 2:

Hardcastle-Kille

[Page 89]

Return-Path: <postmaster@cs.ucl.ac.uk>  
 Received: from cs.ucl.ac.uk by bells.cs.ucl.ac.uk  
   via Delivery Reports Channel id <27718-0@bells.cs.ucl.ac.uk>;  
   Thu, 7 Feb 1991 15:49:11 +0000  
 X400-Received: by mta bells.cs.ucl.ac.uk in  
   /PRMD=uk.ac/ADMD=gold 400/C=gb/;  
   Relayed; Thu, 7 Feb 1991 15:49:08 +0000  
 X400-Received: by /PRMD=DGC/ADMD=GOLD 400/C=GB/; Relayed;  
   Thu, 7 Feb 1991 15:48:40 +0000  
 From: UCL-CS MTA <postmaster@cs.ucl.ac.uk>  
 To: S.Kille@cs.ucl.ac.uk  
 Subject: Delivery Report (failure) for  
   j.nosuchuser@dle.cambridge.DGC.gold-400.gb  
 Message-Type: Delivery Report  
 Date: Thu, 7 Feb 1991 15:49:11 +0000  
 Message-ID: <"DLE/910207154840Z/000"@cs.ucl.ac.uk>  
 Content-Identifier: A useful mess...

This report relates to your message: A useful mess...  
 Your message was not delivered to  
   j.nosuchuser@dle.cambridge.DGC.gold-400.gb  
 for the following reason:  
 Bad Address  
 DG 21187: (CEO POA) Unknown addressee.

\*\*\*\*\* The following information is directed towards the local  
 \*\*\*\*\* administrator and is not intended for the end user  
 \*  
 \* DR generated by /PRMD=DGC/ADMD=GOLD 400/C=GB/  
 \*       at Thu, 7 Feb 1991 15:48:40 +0000  
 \*  
 \* Converted to RFC 822 at bells.cs.ucl.ac.uk  
 \*       at Thu, 7 Feb 1991 15:49:12 +0000  
 \*  
 \* Delivery Report Contents:  
 \*  
 \* Subject-Submission-Identifier:  
 \*   [/PRMD=uk.ac/ADMD=gold 400/C=gb/;<1796.665941626@UK.AC.UCL.CS>]  
 \* Content-Identifier: A useful mess...  
 \* Recipient-Info: j.nosuchuser@dle.cambridge.DGC.gold-400.gb,  
 \*   /I=j/S=nosuchuser/OU=dle/O=cambridge/PRMD=DGC/ADMD=GOLD 400/C=GB/;  
 \*   FAILURE reason Unable-To-Transfer (1);  
 \*   diagnostic Unrecognised-ORName (0);  
 \*   supplementary info "DG 21187: (CEO POA) Unknown addressee.";  
 \*\*\*\*\* End of administration information

The Original Message is not available

### 5.3.9. Probe

This is an MTS internal issue. Any probe shall be serviced by the gateway, as there is no equivalent RFC 822 functionality. The value of the reply is dependent on whether the gateway could service an MTS Message with the values specified in the probe. The reply shall make use of MTS.SupplementaryInformation to indicate that the probe was serviced by the gateway.

### Appendix A - Mappings Specific to SMTP

This Appendix is specific to the Simple Mail Transfer Protocol (RFC 821). It describes specific changes in the context of this protocol. When servicing a probe, as described in section 5.3.9, use may be made of the SMTP VRFY command to increase the accuracy of information contained in the delivery report.

### Appendix B - Mappings specific to the JNT Mail

This Appendix is specific to the JNT Mail Protocol. It describes specific changes in the context of this protocol.

#### 1. Introduction

There are five aspects of a gateway which are JNT Mail Specific. These are each given a section of this appendix.

#### 2. Domain Ordering

When interpreting and generating domains, the UK NRS domain ordering shall be used, both in headers, and in text generated for human description.

#### 3. Addressing

A gateway which maps to JNT Mail should recognise the Domain Defined Attribute JNT-MAIL. The value associated with this attribute should be interpreted according to the JNT Mail Specification. This DDA shall never be generated by a gateway. For this reason, the overflow mechanism is not required.

#### 4. Acknowledge-To:

This field has no direct functional equivalent in X.400. However, it can be supported to an extent, and can be used to improve X.400 support.

If an Acknowledge-To: field is present when going from JNT Mail to

X.400, there are two different situations. The first case is where there is one address in the Acknowledge-To: field, and it is equal to the 822-MTS return address. In this case, the MTS.PerRecipientSubmissionFields.originator-request-report shall be set for each recipient, and the Acknowledge-To: field discarded. Here, X.400 can provide the equivalent service.

In all other cases two actions are taken.

1. Acknowledgement(s) may be generated by the gateway. The text of these acknowledgements shall indicate that they are generated by the gateway, and do not correspond to delivery.
2. The Acknowledge-To: field shall be passed as an extension heading.

When going from X.400 to JNT Mail, in cases where MTA.PerRecipientMessageTransferFields.per-recipient-indicators.originator-report bit is set for all recipients (i.e., there is a user request for a positive delivery report for every recipient), generate an Acknowledge-To: field containing the MTS.OtherMessageDeliveryFields.originator-name. Receipt notification requests are not mapped onto Acknowledge-To:, as no association can be guaranteed between IPMS and MTS level addressing information.

#### 5. Trace

JNT Mail trace uses the Via: syntax. When going from JNT Mail to X.400, a mapping similar to that for Received: is used. No MTS.GlobalDomainIdentifier of the site making the trace can be derived from the Via:, so a value for the gateway is used. The trace text, including the "Via:", is unfolded, truncated to MTS.ub-mta-name-length (32), and mapped to MTA.InternalTraceInformationElement.mta-name. There is no JNT Mail specific mapping for the reverse direction.

#### 6. Timezone specification

The extended syntax of zone defined in the JNT Mail Protocol shall be used in the mapping of UTCTime defined in Chapter 3.

#### 7. Lack of 822-MTS originator specification

In JNT Mail the default mapping of the MTS.OtherMessageDeliveryFields.originator-name is to the Sender: field. This can cause a problem when going from X.400 to JNT Mail if the mapping of IPMS.Heading has already generated a Sender:

field. To overcome this, new extended JNT Mail field is defined. This is chosen to align with the JNT recommendation for interworking with full RFC 822 systems [Kille84b].

original-sender = "Original-Sender" ":" mailbox

If an IPM has no IPMS.Heading.authorizing-users component and IPMS.Heading.originator.formal-name is different from MTS.OtherMessageDeliveryFields.originator-name, map MTS.OtherMessageDeliveryFields.originator-name, onto the Sender: field.

If an IPM has a IPMS.Heading.authorizing-users component, and IPMS.Heading.originator.formal-name is different from MTS.OtherMessageDeliveryFields.originator-name, MTS.OtherMessageDeliveryFields.originator-name is mapped onto the Sender: field, and IPMS.Heading.originator mapped onto the Original-Sender: field.

In other cases the MTS.OtherMessageDeliveryFields.originator-name, is already correctly represented.

#### Appendix C - Mappings specific to UUCP Mail

Gatewaying of UUCP and X.400 is handled by first gatewaying the UUCP address into RFC 822 syntax (using RFC 976) and then gatewaying the resulting RFC 822 address into X.400. For example, an X.400 address

Country	US
Organisation	Xerox
Personal Name	John Smith

might be expressed from UUCP as

inthop!gate!gatehost.COM!/C=US/O=Xerox/PN=John.Smith/

(assuming gate is a UUCP-ARPA gateway and gatehost.COM is an ARPA-X.400 gateway) or

inthop!gate!Xerox.COM!John.Smith

(assuming that Xerox.COM and /C=US/O=Xerox/ are equivalent.)

In the other direction, a UUCP address Smith@ATT.COM, integrated into 822, would be handled as any other 822 address. A non-integrated address such as inthop!dest!user might be handled through a pair of gateways:

Country	US
ADMD	ATT
PRMD	ARPA
Organisation	GateOrg
RFC-822	inthop!dest!user@gatehost.COM

or through a single X.400 to UUCP gateway:

Country	US
ADMD	ATT
PRMD	UUCP
Organisation	GateOrg
RFC-822	inthop!dest!user

#### Appendix D - Object Identifier Assignment

An object identifier is needed for the extension IPMS element. The following value shall be used.

```
rfc-987-88 OBJECT IDENTIFIER ::=
    {ccitt data(9) pss(2342) ucl(234219200300) rfc-987-88(200)}
```

```
id-rfc-822-field-list OBJECT IDENTIFIER ::= {rfc987-88 field(1)}
```

#### Appendix E - BNF Summary

```
boolean = "TRUE" / "FALSE"
```

```
numericstring = *DIGIT
```

```
printablestring = *( ps-char )
ps-restricted-char = 1DIGIT / 1ALPHA / " " / "'" / "+"
                  / ",", / "-" / "." / "/" / ":" / "=" / "?"
ps-delim         = "(" / ")"
ps-char          = ps-delim / ps-restricted-char
```

```
ps-encoded       = *( ps-restricted-char / ps-encoded-char )
ps-encoded-char = " (a) " ; (@)
                  / " (p) " ; (%)
                  / " (b) " ; (!)
                  / " (q) " ; (")
                  / " (u) " ; ( )
                  / " (l) " ; "("
                  / " (r) " ; ")"
                  / " (" 3DIGIT ")"
```

```

teletex-string    = *( ps-char / t61-encoded )
t61-encoded       = "{ " 1* t61-encoded-char "}"
t61-encoded-char  = 3DIGIT

teletex-and-or-ps = [ printablestring ] [ "*" teletex-string ]

labelled-integer ::= [ key-string ] "(" numericstring ")"

key-string        = *key-char
key-char          = <a-z, A-Z, 0-9, and "-">

object-identifier ::= oid-comp object-identifier
                  | oid-comp

oid-comp ::= [ key-string ] "(" numericstring ")"

encoded-info      = 1#encoded-type

encoded-type      = built-in-eit / object-identifier

built-in-eit      = "Undefined"           ; undefined (0)
                  / "Telex"               ; tLX (1)
                  / "IA5-Text"            ; iA5Text (2)
                  / "G3-Fax"              ; g3Fax (3)
                  / "TIF0"                ; tIF0 (4)
                  / "Teletex"             ; tTX (5)
                  / "Videotex"            ; videotex (6)
                  / "Voice"               ; voice (7)
                  / "SFD"                 ; sFD (8)
                  / "TIF1"                ; tIF1 (9)

encoded-pn        = [ given "." ] *( initial "." ) surname

given             = 2*<ps-char not including ".">

initial          = ALPHA

surname          = printablestring

std-or-address    = 1*( "/" attribute "=" value ) "/"
attribute         = standard-type
                  / "RFC-822"
                  / registered-dd-type

```

```

                                / dd-key "." std-printablestring
standard-type                 = key-string

registered-dd-type
                                = key-string
dd-key                        = key-string

value                         = std-printablestring

std-printablestring
                                = *( std-char / std-pair )
std-char                      = <"{", "}", "*", and any ps-char
                                except "/" and "=">
std-pair                      = "$" ps-char

dmn-or-address                = dmn-part *( "." dmn-part )
dmn-part                      = attribute "$" value
attribute                     = standard-type
                                / "~" dmn-printablestring
value                         = dmn-printablestring
                                / "@"
dmn-printablestring =
                                = *( dmn-char / dmn-pair )
dmn-char                      = <"{", "}", "*", and any ps-char
                                except ".">
dmn-pair                      = "\."

global-id = std-or-address


mta-field                     = "X400-Received" ":" x400-trace
                                / "Deferred-Delivery" ":" date-time
                                / "Latest-Delivery-Time" ":" date-time

x400-trace                    = "by" md-and-mta ";"
                                [ "deferred until" date-time ";" ]
                                [ "converted" "(" encoded-info ")" ";" ]
                                [ "attempted" md-or-mta ";" ]
                                action-list
                                ";" arrival-time

md-and-mta                    = [ "mta" mta "in" ] global-id
mta                           = word
arrival-time                   = date-time

```



```

md-or-mta          = "MD" global-id
                    / "MTA" mta

Action-list        = 1#action
action             = "Redirected"
                    / "Expanded"
                    / "Relayed"
                    / "Rerouted"

dr-body-format = dr-summary <CRLF>
                 dr-recipients <CRLF>
                 dr-administrator-info-envelope <CRLF>
                 dr-content-return

dr-content-return = "The Original Message is not available"
                   / "The Original Message follows:"

dr-summary = "This report relates to your message:" <CRLF>
            content-correlator <CRLF> <CRLF>
            "of" date-time <CRLF> <CRLF>

dr-recipients = *(dr-recipient <CRLF> <CRLF>)

dr-recipient = dr-recipient-success / dr-recipient-failure

dr-recipient-success =
    "Your message was successfully delivered to:"
    mailbox "at" date-time

dr-recipient-failure = "Your message was not delivered to:"
                      mailbox <CRLF>
                      "for the following reason:" *word

dr-administrator-info-envelope = 3*( "*" text <CRLF> )

dr-administrator-info =
    "***** The following information is directed towards"
    "the local administrator" <CRLF>
    "***** and is not intended for the end user" <CRLF> <CRLF>
    "DR generated by:" report-point <CRLF>
    "at" date-time <CRLF> <CRLF>
    "Converted to RFC 822 at" mta <CRLF>
    "at" date-time <CRLF> <CRLF>

```

```

"Delivery Report Contents:" <CRLF> <CRLF>
drc-field-list <CRLF>
"***** End of administration information"

drc-field-list          = *(drc-field <CRLF>)

drc-field = "Subject-Submission-Identifier" ":"
            mts-msg-id
            / "Content-Identifier" ":" printablestring
            / "Content-Type" ":" mts-content-type
            / "Original-Encoded-Information-Types" ":"
            encoded-info
            / "Originator-and-DL-Expansion-History" ":"
            dl-history
            / "Reporting-DL-Name" ":" mailbox
            / "Content-Correlator" ":" content-correlator
            / "Recipient-Info" ":" recipient-info
            / "Subject-Intermediate-Trace-Information" ":"
            x400-trace

recipient-info = mailbox "," std-or ";"
report-type
[ "converted eits" encoded-info ";" ]
[ "originally intended recipient"
    mailbox "," std-or ";" ]
[ "last trace" [ encoded-info ] date-time ";" ]
[ "supplementary info" <"> printablestring <"> ";" ]
[ "redirection history" 1#redirection ";" ]
[ "physical forwarding address"
    printablestring ";" ]

report-type          = "SUCCESS" drc-success
                    / "FAILURE" drc-failure

drc-success          = "delivered at" date-time ";"
                    [ "type of MTS user" labelled-integer ";" ]

drc-failure          = "reason" labelled-integer ";"
                    [ "diagnostic" labelled-integer ";" ]

report-point = [ "mta" word "in" ] global-id
content-correlator = *word
dl-history = 1#( mailbox "(" date-time ")" )

```

```

mts-field = "X400-MTS-Identifier" ":" mts-msg-id
           / "X400-Originator" ":" mailbox
           / "X400-Recipients" ":" 1#mailbox
           / "Original-Encoded-Information-Types" ":"
             encoded-info
           / "X400-Content-Type" ":" mts-content-type
           / "Content-Identifier" ":" printablestring
           / "Priority" ":" priority
           / "Originator-Return-Address" ":" 1#mailbox
           / "DL-Expansion-History" ":" mailbox ";" date-time ";"
           / "Conversion" ":" prohibition
           / "Conversion-With-Loss" ":" prohibition
           / "Requested-Delivery-Method" ":"
             1*( labelled-integer )
           / "Delivery-Date" ":" date-time
           / "Discarded-X400-MTS-Extensions" ":"
             1#( oid / labelled-integer )

prohibition      = "Prohibited" / "Allowed"

mts-msg-id       = "[" global-id ";" *text "]"

mts-content-type = "P2" / labelled-integer
                  / object-identifier

priority         = "normal" / "non-urgent" / "urgent"

ipn-body-format  = ipn-description <CRLF>
                  [ ipn-extra-information <CRLF> ]
                  [ ipn-content-return ]

ipn-description  = ipn-receipt / ipn-non-receipt

ipn-receipt      = "Your message to:" preferred-recipient <CRLF>
                  "was received at" receipt-time <CRLF> <CRLF>
                  "This notification was generated"
                  acknowledgement-mode <CRLF>
                  "The following extra information was given:" <CRLF>
                  ipn-suppl <CRLF>

ipn-non-receipt  = "Your message to:"
                  preferred-recipient <CRLF>
                  ipn-reason

ipn-reason       = ipn-discarded / ipn-auto-forwarded

```

```

ipn-discarded = "was discarded for the following reason:"
                discard-reason <CRLF>

ipn-auto-forwarded = "was automatically forwarded." <CRLF>
                    [ "The following comment was made:"
                      auto-comment ]

ipn-extra-information =
    "The following information types were converted:"
    encoded-info

ipn-content-return = "The Original Message is not available"
                    / "The Original Message follows:"
                    <CRLF> <CRLF> message

preferred-recipient = mailbox
receipt-time        = date-time
auto-comment        = printablestring
ipn-suppl           = printablestring

discard-reason      = "Expired" / "Obsoleted" /
                    "User Subscription Terminated"

acknowledgement-mode = "Manually" / "Automatically"

ipms-field = "Obsoletes" ":" 1#msg-id
            / "Expiry-Date" ":" date-time
            / "Reply-By" ":" date-time
            / "Importance" ":" importance
            / "Sensitivity" ":" sensitivity
            / "Autoforwarded" ":" boolean
            / "Incomplete-Copy" ":"
            / "Language" ":" language
            / "Message-Type" ":" message-type
            / "Discarded-X400-IPMS-Extensions" ":" 1#oid

importance          = "low" / "normal" / "high"

sensitivity         = "Personal" / "Private" /
                    "Company-Confidential"

language            = 2*ALPHA [ language-description ]

```

```

language-description = printable-string

message-type      = "Delivery Report"
                  / "InterPersonal Notification"
                  / "Multiple Part"

redirect-comment  =
    [ "Originally To:" ] mailbox "Redirected"
    [ "Again" ] "on" date-time
    "To:" redirection-reason

redirection-reason =
    "Recipient Assigned Alternate Recipient"
    / "Originator Requested Alternate Recipient"
    / "Recipient MD Assigned Alternate Recipient"

subject-line     = "Delivery-Report" "(" status ")"
                  [ "for" destination ]

status           = "success" / "failure" / "success and failures"

destination      = mailbox / "MTA" word

extended-heading =
    "Prevent-NonDelivery-Report" ":"
    / "Generate-Delivery-Report" ":"
    / "Alternate-Recipient" ":" prohibition
    / "Disclose-Recipients" ":" prohibition
    / "Content-Return" ":" prohibition

```

## Appendix F - Format of address mapping tables

### 1. Global Mapping Information

The consistent operation of gateways which follow this specification relies on the existence of three globally defined mappings:

1. Domain Name Space -> O/R Address Space
2. O/R Address Space -> Domain Name Space
3. Domain Name Space -> O/R Address of preferred gateway

All gateways conforming to this specification shall have access to these mappings. The gateway may use standardised or private mechanisms to access this mapping information.

One means of distributing this information is in three files. This appendix defines a format for these files. Other standardised mechanisms to distribute the mapping information are expected. In particular, mechanisms for using the Domain Name Scheme, and X.500 are planned.

The definition of global mapping information is being co-ordinated by the COSINE-MHS project, on behalf of the Internet and other X.400 and RFC 822 users. For information on accessing this information contact:

COSINE MHS Project Team  
SWITCH  
Weinbergstrasse 18  
8001 Zuerich  
Switzerland

tel: +41 1 262 3143

fax: +41 1 262 3151

email:

C=ch;ADMD=arcom;PRMD=switch;O=switch;OU=cosine-mhs;

S=project-team

or

project-team@cosine-mhs.switch.ch

## 2. Syntax Definitions

An address syntax is defined, which is compatible with the syntax used for 822.domains. By representing the O/R addresses as domains, all lookups can be mechanically implemented as domain -> domain mappings. This syntax defined is initially for use in table format, but the syntax is defined in a manner which makes it suitable to be adapted for use with the Domain Name Service. This syntax allows for a general representation of O/R addresses, so that it can be used in other applications. Not all attributes are used in the table formats defined.

To allow the mapping of null attributes to be represented, the pseudo-value "@" (not a printable string character) is used to indicate omission of a level in the hierarchy. This is distinct from the form including the element with no value, although a correct X.400 implementation will interpret both in the same manner.

This syntax is not intended to be handled by users.

```

dmn-or-address  = dmn-part *( "." dmn-part )
dmn-part        = attribute "$" value
attribute       = standard-type
                / "~" dmn-printablestring
value           = dmn-printablestring
                / "@"
dmn-printablestring =
                = *( dmn-char / dmn-pair )
dmn-char        = <"{", "}", "*", and any ps-char
                except ">"
dmn-pair        = "\".
```

An example usage:

```

~ROLE$Big\Chief.ADMMD$ATT.C$US
PRMD$DEC.ADMMD$@.C$US
```

The first example illustrates quoting of a ".", and the second omission of the ADMMD level. There must be a strict ordering of all components in this table, with the most significant components on the RHS. This allows the encoding to be treated as a domain.

Various further restrictions are placed on the usage of dmn-or-address in the address space mapping tables.

1. Only C, ADMMD, PRMD, O, and up to four OUs may be used.
2. No components shall be omitted from this hierarchy, although the hierarchy may terminate at any level. If the mapping is to an omitted component, the "@" syntax is used.
3. Table Lookups

When determining a match, there are aspects which apply to all lookups. Matches are always case independent. The key for all three tables is a domain. The longest possible match shall be obtained. Suppose the table has two entries with the following keys:

```

K.L
J.K.L
```

Domain "A.B.C" will not return any matches. Domain "I.J.K.L" will match the entry "J.K.L:.

#### 4. Domain -> O/R Address format

The BNF is:

```
domain-syntax "#" dmn-or-address "#"
```

Note that the trailing "#" is used for clarity, as the dmn-or-address syntax might lead to values with trailing blanks. Lines starting with "#" are comments.

For example:

```
AC.UK#PRMD$UK\.AC.ADM$GOLD 400.C$GB#
XEROX.COM#O$Xerox.ADM$ATT.C$US#
GMD.DE#O$@.PRMD$GMD.ADM$DBP.C$DE#
```

A domain is looked up to determine the top levels of an O/R Address. Components of the domain which are not matched are used to build the remainder of the O/R address, as described in Section 4.3.4.

#### 5. O/R Address -> Domain format

The syntax of this table is:

```
dmn-or-address "#" domain-syntax "#"
```

For example:

```
#
# Mapping table
#
PRMD$UK\.AC.ADM$GOLD 400.C$GB#AC.UK#
```

The O/R Address is used to generate a domain key. It is important to order the components correctly, and to fill in missing components in the hierarchy. Use of this mapping is described in Section 4.3.2.

#### 6. Domain -> O/R Address of Gateway table

This uses the same format as the domain -> O/R address mapping. In this case, the two restrictions (omitted components and restrictions on components) do not apply. Use of this mapping is described in Section 4.3.4.



## Appendix G - Mapping with X.400(1984)

This appendix defines modification to the mapping for use with X.400(1984).

The X.400(1984) protocols are a proper subset of X.400(1988). When mapping from X.400(1984) to RFC 822, no changes to this specification are needed.

When mapping from RFC 822 to X.400(1984), no use can be made of 1988 specific features. No use of such features is made at the MTS level. One feature is used at the IPMS level, and this must be replaced by the RFC 987 approach. All header information which would usually be mapped into the rfc-822-heading-list extension, together with any Comments: field in the RFC 822 header is mapped into a single IA5 body part, which is the first body part in the message. This body part will start with the string "RFC-822-Headers:" as the first line. The headers then follow this line. This specification requires correct reverse mapping of this format, either from 1988 or 1984.

In an environment where RFC 822 is of major importance, it may be desirable for downgrading to consider the case where the message was originated in an RFC 822 system, and mapped according to this specification. The rfc-822-heading-list extension may be mapped according to this appendix.

When parsing std-or, the following restrictions must be observed:

- Only the 84/88 attributes identified in the table in Section 4.2 are present.
- No teletex encoding is allowed.

If an address violates this, it should be treated as an RFC 822 address, which will usually lead to encoding as a DDA "RFC-822".

It is possible that null attributes may be present in an O/R Address. This is not legal in 1988, except for ADMD where the case is explicitly described in Section 4.3.5. Null attributes are deprecated (the attribute should be omitted), and should therefore be unusual. However, some systems generate them and rely on them. Therefore, any null attribute shall be encoded using the std-or encoding (e.g., /O=/).

If a non-Teletex Common Name (CN) is present, it should be mapped onto a Domain Defined Attribute "Common". This is in line with RFC 1328 on X.400 1988 to 1984 downgrading [Hardcastle-K92].

## Appendix H - RFC 822 Extensions for X.400 access

This appendix defines a number of optional mappings which may be provided to give access from RFC 822 to a number of X.400 services. These mappings are beyond the basic scope of this specification. There has been a definite demand to use extended RFC 822 as a mechanism to access X.400, and these extensions provide access to certain features. If this functionality is provided, this appendix shall be followed. The following headings are defined:

```
extended-heading =  
    "Prevent-NonDelivery-Report" ":"  
    / "Generate-Delivery-Report" ":"  
    / "Alternate-Recipient" ":" prohibition  
    / "Disclose-Recipients" ":" prohibition  
    / "Content-Return" ":" prohibition
```

Prevent-NonDelivery-Report and Generate-Delivery-Report allow setting of MTS.PerRecipientSubmissionFields.originator-report-request. The setting will be the same for all recipients.

Alternate-Recipient, Disclose-Recipients, and Content-Return allow for override of the default settings for MTS.PerMessageIndicators.

## Appendix I - Conformance

This appendix defines a number of options, which a conforming gateway should specify. Conformance to this specification shall not be claimed if any of the mandatory features are not implemented. In particular:

- Formats for all fields shall be followed.
- Formats for subject lines, delivery reports and IPNs shall be followed. A system which followed the syntax, but translated text into a language other than english would be conformant.
- RFC 1137 shall not be followed when mapping to SMTP or to JNT Mail
- All mappings of trace shall be implemented.
- There must be a mechanism to access all three global mappings.

A gateway should specify:

- Which 822-MTS protocols are supported. The relevant appendices must be followed to claim support of a given protocol: SMTP (A); JNT Mail (B); UUCP (C).
- Which X.400 versions are supported (84 and/or 88).
- The means by which it can access the global mappings. Currently, the tables of the formats define in Appendix F is the only means available.
- The approach taken when upper bounds are exceeded at the IPM level (5.1.3)
- The approach taken to return of contents (5.2)
- The approach taken to body parts which cannot be converted (5.3.4)
- The approach taken to multiple copies vs non-disclosure (4.6.2.2)

The following are optional parts of this specification. A conforming implementation should specify which of these it supports.

- Generation of extended RFC 822 fields is mandatory. Optionally, they may be parsed and mapped back to X.400. A gateway should indicate if this is done.
- Support for the extension mappings of Appendix H.
- Support for returning illegal format content in a delivery report
- Which address interpretation heuristics are supported (4.3.4.1)
- If RFC 987 generated message ids are handled in a backwards compatible manner (4.7.3.6)

#### Appendix J - Change History: RFC 987, 1026, 1138, 1148

RFC 987 was the original document, and contained the key elements of this specification. It was specific to X.400(1984). RFC 1026 specified a small number of necessary changes to RFC 987.

RFC 1138 was based on the RFC 987 work. It contained an editorial error, and was reissued a few months later as RFC 1148. RFC 1148 will be referred to here, as it is the document which is widely

referred to elsewhere. The major goal of RFC 1148 was to upgrade RFC 987 to X.400(1988). It did this, but did not obsolete RFC 987, which was recommended for use with X.400(1984). This appendix summarises the changes made in going from RFC 987 to RFC 1148.

RFC 1148 noted the following about its upgrade from RFC 987: Unnecessary change is usually a bad idea. Changes on the RFC 822 side are avoided as far as possible, so that RFC 822 users do not see arbitrary differences between systems conforming to this specification, and those following RFC 987. Changes on the X.400 side are minimised, but are more acceptable, due to the mapping onto a new set of services and protocols.

## 1. Introduction

The model has shifted from a protocol based mapping to a service based mapping. This has increased the generality of the specification, and improved the model. This change affects the entire document.

A restriction on scope has been added.

## 2. Service Elements

- The new service elements of X.400 are dealt with.
- A clear distinction is made between origination and reception

## 3. Basic Mappings

- Add teletex support
- Add object identifier support
- Add labelled integer support
- Make PrintableString <-> ASCII mapping reversible
- The printable string mapping is aligned to the NBS mapping derived from RFC 987.

## 4. Addressing

- Support for new addressing attributes
- The message ID mapping is changed to not be table driven

## 5. Detailed Mappings

- Define extended IPM Header, and use instead of second body part for RFC 822 extensions
- Realignment of element names
- New syntax for reports, simplifying the header and introducing a mandatory body format (the RFC 987 header format was unusable)
- Drop complex autoforwarded mapping
- Add full mapping for IP Notifications, defining a body format
- Adopt an MTS Identifier syntax in line with the O/R Address syntax
- A new format for X400 Trace representation on the RFC 822 side

## 6. Appendices

- Move Appendix on restricted 822 mappings to a separate RFC
- Delete Phonenet and SMTP Appendixes

## Appendix K - Change History: RFC 1148 to this Document

### 1. General

- The scope of the document was changed to cover X.400(1984), and so obsolete RFC 987.
- Changes were made to allow usage to connect RFC 822 networks using X.400
- Text was tightened to be clear about optional and mandatory aspects
- A good deal of clarification
- A number of minor EBNF errors
- Better examples are given
- Further X.400 upper bounds are handled correctly

## 2. Basic Mappings

- The encoding of object identifier is changed slightly

## 3. Addressing

- A global mapping of domain to preferred gateway is introduced.
- An overflow mechanism is defined for RFC 822 addresses of greater than 128 bytes.
- Changes were made to improve compatability with the PDAM on writing O/R Addresses.
- + The PD and Terminal Type keywords were aligned to the PDAM. It is believed that minimal use has been made of the RFC 1148 keywords.
- + P and A are allowed as alternate keys for PRMD and ADMD
- + Where keywords are different, the PDAM keywords are alternatives on input. This is mandatory.

## 4. Detailed Mappings

- The format of the Subject: lines is defined.
- Illegal use (repetition) of the heading EXTENSION is corrected, and a new object identifier assigned.
- The Delivery Report format is extensively revised in light of operational experience.
- The handling of redirects is significantly changed, as the previous mechanism did not work.

## 5. Appendices

- An SMTP appendix is added, allowing optional use of the VRFY command to improve probe information.
- Handling of JNT Mail Acknowledge-To is changed slightly.
- A DDA JNT-MAIL is allowed on input.
- The format definitions of Appendix F are explained further, and a third table definition added.

- An appendix on use with X.400(1984) is added.
- Optional extensions are defined to give RFC 822 access to further X.400 facilities.
- An appendix on conformance is added.

## References

### CCITT88a.

CCITT, "CCITT Recommendations X.408," Message Handling Systems: Encoded Information Type Conversion Rules, December 1988.

### CCITT/ISO88a.

CCITT/ISO, "CCITT Recommendations X.400/ ISO IS 10021-1," Message Handling: System and Service Overview , December 1988.

### CCITT/ISO88b.

CCITT/ISO, "CCITT Recommendations X.420/ ISO IS 10021-7," Message Handling Systems: Interpersonal Messaging System, December 1988.

### CCITT/ISO88c.

CCITT/ISO, "CCITT Recommendations X.411/ ISO IS 10021-4," Message Handling Systems: Message Transfer System: Abstract Service Definition and Procedures, December 1988.

### CCITT/ISO88d.

CCITT/ISO, "Specification of Abstract Syntax Notation One (ASN.1)," CCITT Recommendation X.208 / ISO IS 8824, December 1988.

### CCITT/ISO91a.

CCITT/ISO, "Representation of O/R Addresses for Human Usage," PDAM to CCITT X.401 / ISO/IEC 10021-2, February 1991.

### Crocker82a.

Crocker, D., "Standard of the Format of ARPA Internet Text Messages," RFC 822, UDEL, August 1982.

### Hardcastle-K92.

Hardcastle-Kille, S., "X.400 1988 to 1984 downgrading," RFC 1328, UCL, May 1992.

Horton86a.

Horton, M., "UUCP Mail Interchange Format Standard," RFC 976, February 1986.

Kille84b.

Kille, S., "Gatewaying between RFC 822 and JNT Mail," JNT Mailgroup Note 15, May 1984.

Kille84a.

Kille, S., (Editor), JNT Mail Protocol (revision 1.0), Joint Network Team, Rutherford Appleton Laboratory, March 1984.

Kille86a.

Kille, S., "Mapping Between X.400 and RFC 822," UK Academic Community Report (MG.19) / RFC 987, June 1986.

Kille87a.

Kille, S., "Addendum to RFC 987," UK Academic Community Report (MG.23) / RFC 1026, August 1987.

Kille89a.

Kille, S., "A String Encoding of Presentation Address," UCL Research Note 89/14, March 1989.

Kille89b.

Kille, S., "Mapping between full RFC 822 and RFC 822 with restricted encoding," RFC 1137, October 1989.

Kille90a.

Kille, S., "Mapping Between X.400(1988) / ISO 10021 and RFC 822," RFC 1148, March 1990.

Larmouth83a.

Larmouth, J., "JNT Name Registration Technical Guide," Salford University Computer Centre, April 1983.

Postel84a.

Postel J., and J. Reynolds, "Domain Requirements," RFC 920, USC/Information Sciences Institute, October 1984.

Postel82a.

Postel, J., "Simple Mail Transfer Protocol", RFC 821, USC/Information Sciences Institute, August 1982.

Rose85a.

Rose M., and E. Stefferud, "Proposed Standard for Message Encapsulation," RFC 934, January 1985.



Systems85a.

CEN/CENELEC/Information Technology/Working Group on Private  
Message Handling Systems, "FUNCTIONAL STANDARD A/3222,"  
CEN/CLC/IT/WG/PMHS N 17, October 1985.

#### SECURITY CONSIDERATIONS

Security issues are not discussed in this memo.

#### AUTHOR'S ADDRESS

Steve Hardcastle-Kille  
Department of Computer Science  
University College London  
Gower Street  
WC1E 6BT  
England

Phone: +44-71-380-7294  
EMail: S.Kille@CS.UCL.AC.UK