

## Equivalences between 1988 X.400 and RFC-822 Message Bodies

### 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.

### Table of Contents

1. Introduction .....	2
2. Equivalence Table Definition .....	2
3. Generic conversions .....	3
3.1. Byte copy .....	3
3.2. Text Conversion .....	3
3.3. Image Conversion .....	3
3.4. Tunneling .....	3
4. Conversion Table for known X.400 and MIME Types .....	4
4.1. MIME to X.400 Table .....	4
4.2. X.400 to MIME Table .....	4
5. Newly defined X.400 Body Parts .....	5
5.1. Use of OBJECT IDENTIFIERS and ASN.1 MACROS .....	5
5.2. The Generic MIME Extended Body Part .....	6
5.3. The PostScript body part .....	7
5.4. The JPEG body part .....	7
5.5. The GIF body part .....	8
6. Newly defined MIME content-types .....	8
6.1. The application/x400-bp content-type .....	8
6.2. The image/g3fax content-type .....	9
6.2.1. G3Fax Parameters .....	9
6.2.2. Content Encoding .....	10
6.3. The Application/ODA content-type .....	11
7. Equivalence Definitions .....	11
7.1. IA5Text - text/plain .....	11
7.2. GeneralText - text/plain (ISO-8859) .....	12
7.3. BilaterallyDefined - application/octet-stream .....	13
7.4. ODA - application/oda .....	14
7.5. g3-facsimile - image/g3fax .....	15
7.6. application/postscript - postscript-body-part .....	16
7.7. application/jpeg - jpeg-body-part .....	16

7.8. image/gif - gif-body-part .....	16
8. OID Assignments .....	17
9. IANA Registration form for new mappings .....	17
10. Security Considerations .....	18
11. Authors' Addresses .....	18
12. References .....	19

## 1. Introduction

This document is a companion to [1], which defines the principles behind interworking between MIME-based RFC-822 mail and X.400 mail. This document describes the content of the "IANA MHS/MIME Equivalence table" referenced in the companion document, and defines the initial configuration of this table. Mappings for new MIME content-types and/or X.400 body part types should be registered with the IANA to minimize redundancy and promote interoperability.

In MIME, the term "content-type" is used to refer to an information object contained in the body of a message. In contrast, X.400 uses the term "body part type." In this document, the term "body part" is used to refer to either.

Please send comments to the MIME-MHS mailing list:  
<mime-mhs@surfnet.nl>.

## 2. Equivalence Table Definition

For each MIME content-type/X.400 body part pair, the Equivalence Table will contain an entry with the following sections:

### X.400 Body Part

This section identifies the X.400 Body Part governed by this Table entry. It includes any OBJECT IDENTIFIERS or other parameters necessary to uniquely identify the Body Part.

### MIME Content-Type

This section identifies the MIME content-type governed by this Table entry. The MIME content-type named here must be registered with the IANA.

### Conversion Type

This section identifies the type of conversion applied. See the section on Generic Conversions for an explanation of the possible values.

#### Comments (optional)

This section gives any additional commentary that might be useful in understanding the mapping between the X.400 and MIME representations.

The initial Equivalence Table entries in this document are described using this convention. Any future submissions to the IANA should follow this format.

### 3. Generic conversions

#### 3.1. Byte copy

This is the trivial case, that is, no conversion at all. The byte stream is simply copied between MIME and X.400.

This is the preferred conversion, since it is the simplest.

Implementors and vendors will be registering OBJECT IDENTIFIERS and MIME content-types for their various objects. They are STRONGLY ENCOURAGED to specify their content formats such that a gateway can use Byte Copy to map between them.

Note that in some cases, it is necessary to define exactly which ASN.1 construct to replace with the content of the MIME object.

#### 3.2. Text Conversion

This type of conversion applies to text objects that cannot be mapped using a simple Byte Copy. Conversion involves scanning and reformatting the object. For example, the MIME and X.400 objects might differ in their encoding of nonstandard characters, or line or page breaks.

#### 3.3. Image Conversion

This conversion type applies to raster images, like Group 3 Facsimile or JPEG. Again, it differs from Byte Copy in that it involves scanning reformatting the byte stream. It differs from Text Conversion in that it is pixel- oriented, rather than character-oriented.

#### 3.4. Tunneling

This is not a conversion at all, but an encapsulation of the object. This is the fallback conversion, used when no explicit mapping applies.

#### 4. Conversion Table for known X.400 and MIME Types

This section itemizes the equivalences for all currently known MIME content-types and X.400 body parts.

##### 4.1. MIME to X.400 Table

MIME content-type -----	X.400 Body Part -----	Section -----
text/plain		
charset=us-ascii	ia5-text	7.1
charset=iso-8859-x	EBP - GeneralText	7.2
text/richtext	no mapping defined	5.2
application/oda	EBP - ODA	7.4
application/octet-stream	bilaterally-defined	7.3
application/postscript	EBP - mime-postscript-body	5.4, 7.6
image/g3fax	g3-facsimile	6.2, 7.5
image/jpeg	EBP - mime-jpeg-body	5.5, 7.7
image/gif	EBP - mime-gif-body	5.6, 7.8
audio/basic	no mapping defined	5.2
video/mpeg	no mapping defined	5.2

Abbreviation: EBP - Extended Body Part

##### 4.2. X.400 to MIME Table

###### Basic Body Parts

X.400 Basic Body Part -----	MIME content-type -----	Section -----
ia5-text	text/plain; charset=us-ascii	7.1
voice	No Mapping Defined	6.1
g3-facsimile	image/g3fax	6.2, 7.5
g4-class1	no mapping defined	6.1
teletex	no mapping defined	6.1
videotex	no mapping defined	6.1
encrypted	no mapping defined	6.1
bilaterally-defined	application/octet-stream	7.3
nationally-defined	no mapping defined	6.1
externally-defined	See Extended Body Parts	6.1

X.400 Extended Body Part -----	MIME content-type -----	Section -----
GeneralText	text/plain; charset=iso-8859-x	7.2
ODA	application/oda	7.4
mime-postscript-body	application/postscript	5.3, 7.6
mime-jpeg-body	image/jpeg	5.4, 7.7
mime-gif-body	image/gif	5.5, 7.8

## 5. Newly defined X.400 Body Parts

This section defines new X.400 Body Parts for the purposes of interworking with MIME.

All new X.400 Body Parts defined here will be Extended Body Parts, as defined in CCITT Recommendation X.420 [2].

### 5.1. Use of OBJECT IDENTIFIERS and ASN.1 MACROS

X.420 dictates that Extended Body Parts shall:

- (1) use OBJECT IDENTIFIERS (OIDs) to uniquely identify the contents, and
- (2) be defined by using the ASN.1 Macro:

```
EXTENDED-BODY-PART-TYPE MACRO ::=
BEGIN
    TYPE NOTATION    ::= Parameters Data
    VALUE NOTATION   ::= value (VALUE OBJECT IDENTIFIER)

    Parameters       ::=  "PARAMETERS" type "IDENTIFIED"
                        |  "BY" value(OBJECT IDENTIFIER)
                        |  empty;
    Data              ::= "DATA" type
END
```

To meet these requirements, this document uses the OID

mime-mhs-bodies

defined in [1], as the root OID for X.400 Extended Body Parts defined for MIME interworking.

Each Extended Body Part contains Data and optional Parameters, each being named by an OID. To this end, two OID subtrees are defined under mime-mhs-bodies, one for Data, and the other for Parameters:

```
mime-mhs-bp-data OBJECT IDENTIFIER ::=
    { mime-mhs-bodies 1 }
mime-mhs-bp-parameter OBJECT IDENTIFIER ::=
    { mime-mhs-bodies 2 }
```

All definitions of X.400 body parts submitted to the IANA for registration must use the Extended Body Part Type macro for the definition. See the next section for an example.

Lastly, the IANA will use the mime-mhs-bp-data and mime-mhs-bp-parameter OIDs as root OIDs for any new MIME content-type/subtypes that aren't otherwise registered in the Equivalence Table.

## 5.2. The Generic MIME Extended Body Part

The following X.400 Body Part is defined to carry any MIME content-type for which there is no explicit IANA registered mapping.

```

mime-body-part EXTENDED-BODY-PART-TYPE
  PARAMETERS MimeParameters
    IDENTIFIED BY mime-generic-parameters
  DATA          OCTET STRING
  ::= mime-generic-data

MimeParameters ::=
  SEQUENCE {
    content-type          IA5String,
    content-parameters SEQUENCE OF
      SEQUENCE {
        parameter          IA5String,
        parameter-value    IA5String
      }
    -- from RFC-1327, sec. 5.1.12
    other-header-fields RFC822FieldList
  }

mime-generic-parameters OBJECT IDENTIFIER ::=
  { mime-mhs-bp-parameter 1 }
mime-generic-data        OBJECT IDENTIFIER ::=
  { mime-mhs-bp-data 1 }

```

To convert the MIME content-type into the X.400 mime- body-part:

- (1) Copy the "type/subtype" string from the MIME Content-Type: header field into MimeParameters.content-type
- (2) For each "parameter=value" string in the MIME Content-Type header field, create a MimeParameters.content-parameters structure, and copy the "parameter" string into MimeParameters.content-parameters.parameter field and the "value" string into the paired MimeParameters.content-parameters.parameter-value field.
- (3) Convert the MIME body part into its canonical form.

(See appendix H of RFC 1341 [3] for a discussion of canonical in this context.) Said another way, reverse the transfer encoding to recover the original byte stream.

- (4) Copy the canonical byte stream into the mime-body-part.data octet string.
- (5) Remove the Content-type and the Content-transfer-encoding header fields from the MIME body part's RFC822 header.
- (6) Any header fields starting with "Content-" in the MIME body part is placed in the optional other-header-fields structure. Note that this can only occur when the MIME content-type occurs as part of a "multipart" content-type.

The mapping from the X.400 mime-body-part to a MIME content-type is the inverse of the above steps.

### 5.3. The PostScript body part

The following Extended Body Part is defined for PostScript data streams. It has no parameters.

```
postscript-body-part EXTENDED-BODY-PART-TYPE
```

```
DATA OCTET STRING
::= mime-postscript-body
```

```
mime-postscript-body OBJECT IDENTIFIER ::=
    { mime-mhs-bp-data 2 }
```

### 5.4. The JPEG body part

The following Extended Body Part is defined for JPEG data streams. It has no parameters.

```
jpeg-body-part EXTENDED-BODY-PART-TYPE
DATA OCTET STRING
::= mime-jpeg-body
```

```
mime-jpeg-body OBJECT IDENTIFIER ::=
    { mime-mhs-bp-data 3 }
```

### 5.5. The GIF body part

The following Extended Body Part is defined for GIF data streams. It has no parameters.

```
gif-body-part EXTENDED-BODY-PART-TYPE
  DATA          OCTET STRING
  ::= mime-gif-body
```

```
mime-gif-body OBJECT IDENTIFIER ::=
  { mime-mhs-bp-data 4 }
```

## 6. Newly defined MIME content-types

This section defines new MIME content-types for the purposes of interworking with X.400.

### 6.1. The application/x400-bp content-type

This content-type is defined to carry any X.400(88) body part for which there is no registered IANA mapping.

The content-type field is

```
application/x400-bp
```

The parameters are:

```
bp-type=<INTEGER or OBJECT IDENTIFIER>
```

The body contains the raw ASN.1 IPM.body octet stream, including the initial tag octet.

If the body is a basic body part, the bp-type parameter is set to the number of the body part's context-specific tag, that is, the tag of the IPMS.Body.BodyPart component.

If the body is an Extended Body Part, the bp-type parameter is set to the OBJECT IDENTIFIER from

```
IPMS.body.externally-defined.data.direct-reference
```

No attempt is made to turn the parameters of Extended Body Parts into MIME parameters. (This task is the responsibility of the recipient's UA).

For example, a basic VideotexBodyPart will have



Content-type=application/x400-bp; bp-type=6

whilst a Extended Videotex body part will have

Content-type=application/x400-bp; bp-type=2.6.1.4.5

application/x400-bp will need a content-transfer-encoding of base64 or quoted-printable when carried in 7-bit MIME. Since there is no way to know beforehand the content, it is recommended to just inspect the first 1 KByte or so of data and choose the one that seems to produce the more compact encoding.

If this is not feasible, Base64 is recommended.

## 6.2. The image/g3fax content-type

This content-type is defined to carry G3 Facsimile byte streams.

In general, a G3Fax image contains 3 pieces of information:

- (1) A set of flags indicating the particular coding scheme. CCITT Recommendation T.30 defines how the flags are transmitted over telephones. In this medium, the flags are carried as parameters in the MIME content-type header field.
- (2) A structure that divides the bits into pages. CCITT recommendation T.30 describes how to define page boundaries. A page break algorithm is defined here that is independent of how the image data is conveyed.
- (3) For each page, a sequence of bits that form the encoding of the image. CCITT recommendation T.4 defines the bit image format. This is used without change.

### 6.2.1. G3Fax Parameters

The following parameters are defined:

- (1) page-length - possible values: A4, B4 and Unlimited
- (2) page-width - possible values: A3, A4, B4
- (3) encoding - possible values: 1-dimensional, 2-dimensional, Uncompressed

- (4) resolution - possible values: Fine, Coarse
- (5) DCS - a bit string, represented in Base64.
- (6) pages - an integer, giving the number of pages in the document

If nothing is specified, the default parameter settings are:

```

page-length=A4
page-width=A4
encoding=1-dimensional
resolution=Coarse

```

It is possible (but misleading) to view the representation of these values as single-bit flags. They correspond to the following bits of the T.30 control string and X.400 G3FacsimileParameters:

Parameter	T.30 bit	X.400 bit
page-length=A4	no bit set	
page-length=B4	19	21
page-length=Unlimited	20	20
page-width=A4	no bit set	
page-width=A3	18	22
page-width=B4	17	23
encoding=1-dimensional	no bit set	
encoding=2-dimensional	16	8
encoding=Uncompressed	26	30
resolution=Coarse	no bit set	
resolution=Fine	15	9

The reason for the different bit numbers is that X.400 counts bits in an octet from the MSB down to the LSB, while T.30 uses the opposite numbering scheme.

If any bit but these are set in the Device Control String, the DCS parameter should be supplied.

#### 6.2.2. Content Encoding

X.400 defines the g3-facsimile data stream as a SEQUENCE of BIT STRINGS. Each BIT STRING is a page of facsimile image data, encoded as defined by Recommendation T.4. The following content encoding is reversible between MIME and X.400 and ensures that page breaks are

honored in the MIME representation.

An EOL is defined as a bit sequence of

000000000001 (eleven zeroes and a one).

Each page of the message is delimited by a sequence of six (6) EOLs that MUST start on a byte boundary. The image bit stream is padded as needed to achieve this alignment.

Searching for the boundary is a matter of searching for the byte sequence (HEX) 00 10 01 00 10 01 00 10 01, which cannot occur inside the image.

See Section 7.5 for the algorithm on conversion between this encoding and the X.400 encoding.

The Base64 content-transfer-encoding is appropriate for carrying this content-type.

### 6.3. The Application/ODA content-type

The "ODA" subtype of application is used to indicate that a body contains information encoded according to the Office Document Architecture [4] standards, using the ODIF representation format. For application/oda, the Content-Type line should also specify an attribute/value pair that indicates the document application profile (DAP), using the key word "profile", and the document class, using the keyword "class".

For the keyword "class", the values "formatted", "processable" and "formatted-processable" are legal values.

Thus an appropriate header field might look like this:

```
Content-Type: application/oda; profile=Q112;
class=formatted
```

Consult the ODA standard [4] for further information.

The Base64 content-transfer-encoding is appropriate for carrying ODA.

## 7. Equivalence Definitions

### 7.1. IA5Text - text/plain

X.400 Body Part: IA5Text  
MIME Content-type: text/plain; charset=US-ASCII

Conversion Type: Byte copy  
Comments:

When mapping from X.400 to MIME, the "repertoire" parameter is ignored.

When mapping from MIME to X.400, the "repertoire" parameter is set to IA5 (5).

NOTE: The MIME Content-type headers are omitted, when mapping from X.400 to MIME, if and only if the IA5Text body part is the only body part in the IPMS.Body sequence.

NOTE: IA5Text specifies the "currency" symbol in position 2/4. This is converted without comment to the "dollar" symbol, since the author of this document has seen many documents in which the position was intended to indicate "dollar" while he has not yet seen one in which the "currency" symbol is intended.

(For reference: The T.50 (1988) recommendation, which defines IA5, talks about ISO registered set number 2, while ASCII, using the "dollar" symbol, is ISO registered set number 6. There are no other differences.)

## 7.2. GeneralText - text/plain (ISO-8859)

X.400 Body Part: GeneralText; CharacterSets in  
6,100,101,109,110,126,127,138,144,148  
MIME Content-Type: text/plain; charset=ISO-8859-(1-9)  
Conversion Type: Byte copy  
Comments:

When mapping from X.400 to MIME, the character-set chosen from table below according to the value of Parameters.CharacterSets.

When mapping from MIME to X.400, GeneralText is an Extended Body Part, hence it requires an OID. The OID for the GeneralText body is defined in [5], part 8, annex D, as {2 6 1 4 11}. The OID for the parameters is {2 6 1 11 11}.

The Parameters.CharacterSets is set from table below according to the value of "charset"

NOTE: The GeneralText body part is defined in ISO 10021-8 [5], and NOT in the corresponding CCITT recommendation. Its parameters were heavily modified in a defect report, and will be a SET OF INTEGER (indicating the ISO registry numbers of all the used sets) in the next version of the standard.

The following table lists the MIME character sets and the corresponding ISO registry numbers. If no correspondence is found, this conversion fails, and the generic body part approach is used.

MIME charset	ISO IR numbers	Comment
-----	-----	-----
ISO-8859-1	6, 100	West European "8-bit ASCII"
ISO-8859-2	6, 101	East European
ISO-8859-3	6, 109	<regarded as obsolete>
ISO-8859-4	6, 110	<regarded as obsolete>
ISO-8859-5	6, 144	Cyrillic
ISO-8859-6	6, 127	Arabic
ISO-8859-7	6, 126	Greek
ISO-8859-8	6, 138	Hebrew
ISO-8859-8	6, 148	Other Latin-using languages

When converting from MIME to X.400, generate the correct OIDs for use in the message envelope's Encoded Information Types by looking up the ISO IR number in the above table, and then appending it to the id-cs-eit-authority {1 0 10021 7 1 0} OID.

The escape sequences to designate and invoke the relevant character sets in their proper positions must be added to the front of the GeneralText character string.

### 7.3. BilaterallyDefined - application/octet-stream

X.400 Body Part: BilaterallyDefined  
 MIME Content-Type: Application/Octet-Stream (no parameters)  
 Conversion Type: Byte copy  
 Comments:

When mapping from MIME to X.400, if there are parameters present in the Content-Type: header field, the conversion fails since the BilaterallyDefined Body Part does not have any corresponding ASN.1 parameters.

DISCUSSION: The parameters "name" "type" and "conversions" are advisory, but may in some cases give vital hints on the expected handling of the file. The parameter "conversions" is not fully defined, but it is expected that it will be useful, so we cannot drop it and expect people to be satisfied.

The parameter "padding" changes the interpretation of the last byte of the data, and so cannot be deleted.

An option is to prepend an IA5 body part that contains the parameter text; this will aid unmodified readers, and can probably be made

reversible with suitable chicanery, but is it worth it????

Also, use of BilaterallyDefined Body Parts is specifically deprecated in both 1988 and 1992 X.400. It is retained solely for backward compatibility with 1984 systems. 1992 X.400 defines a File Transfer Body Part to solve this problem (i.e. binary file transfer through email). The standard and its regional profiles are not solid enough yet to exploit as a solution for this problem.

#### 7.4. ODA - application/oda

X.400 Body Part: ODA  
 MIME Content-Type: application/oda  
 Conversion Type: Byte copy  
 Comments:

The ODA body part is defined in the CCITT document T.411 [6], appendix E, section E.2, "ODA identification in the P2 protocol of MHS"

An abbreviated version of its ASN.1 definition is:

```
oda-body-part EXTENDED-BODY-PART-TYPE
    PARAMETERS      OdaBodyPartParameters
    DATA           OdaData
    ::= id-et-oda

OdaBodyPartParameters ::= SET {
    document-application-profile    [0] OBJECT IDENTIFIER
    document-architecture-class    [1] INTEGER {
                                     formatted (0)
                                     processable (1)
                                     formatted-processable(2)} }

id-et-oda OBJECT IDENTIFIER ::= { 2 8 1 0 1 }
```

Mapping from X.400 to MIME, the following is done:

The Parameters.document-application-profile is mapped onto the MIME parameter "profile" according to the table below.

Profile	OBJECT IDENTIFIER
Q112	{ iso (1) identified-organization (3) ewos (16) eg (2) oda (6) profile (0) q112 (1) }

The Parameters.document-architecture-class is mapped onto the MIME parameter "class" according to the table below

String	Integer
formatted	formatted(0)
processable	processable(1)
formatted-processable	formatted-processable(2)

NOTE: This parameter is not defined in RFC 1341.

The body of the MIME content-type is the Data part of the ODA body part.

When mapping from MIME to X.400, the following steps are done:

The Parameters.document-application-profile and Parameters.document-architecture-class are set from the tables above. If any of the parameters are missing, the values for Q112 and formatted-processable are used.

It is an option for the gateway implementor to try to access them from inside the document, where they are defined as

document-profile.document-characteristics.document-architecture-class

document-profile.document-characteristics.document-application-profile

Gateways are NOT required to do this, since the document-characteristics are optional parameters. If a gateway does not, it simply uses the defaulting rules defined above.

The OBJECT IDENTIFIERS for the document application profile and for ODA {2 8 0 0} must be added to the Encoded Information Types parameter of the message envelope.

#### 7.5. g3-facsimile - image/g3fax

X.400 Body part: g3-facsimile  
 MIME Content-Type: image/g3fax  
 Conversion Type: nearly Byte copy  
 Comments:

The Parameters of the X.400 G3Fax body part are mapped to the corresponding Parameters on the MIME Image/G3Fax body part and vice versa. Note that:

- (1) If fineResolution is not specified, pixels will be twice as tall as they are wide
- (2) If any bit not corresponding to a specially named

option is set in the G3Fax NonBasicParameters, the "DCS" parameter must be used.

- (3) Interworking is not guaranteed if any bit apart from those specially named are used in the NonBasicParameters

From X.400 to G3Fax, the body is created in the following way:

- (1) Any trailing EOL markers on each bitstring is removed. The bitstring is padded to a byte boundary.
- (2) 6 consecutive EOL markers are appended to each bitstring.
- (3) The padded bitstrings are concatenated together

An EOL marker is the bit sequence 000000000001 (11 zeroes and a one).

From G3Fax to X.400, the body is created in the following way:

- (1) The body is split into bitstrings at each occurrence of 6 consecutive EOL markers, and trailing EOLs and padding are removed
- (2) Each bitstring is made into an ASN.1 BITSTRING
- (3) The bitstrings are made into an ASN.1 SEQUENCE, which forms the body of the G3Fax body part.

#### 7.6. application/postscript - postscript-body-part

X.400 Body Part: Extended Body Part, OID postscript-body-part  
MIME Content-Type: application/postscript  
Conversion Type: Byte Copy

#### 7.7. application/jpeg - jpeg-body-part

X.400 Body Part: Extended Body Part, OID jpeg-body-part  
MIME Content-Type: application/jpeg  
Conversion Type: Byte Copy

#### 7.8. image/gif - gif-body-part

X.400 Body Part: Extended Body Part, OID gif-body-part  
MIME Content-Type: application/gif  
Conversion Type: Byte Copy



## 8. OID Assignments

```
MIME-MHS-MAPPINGS DEFINITIONS ::= BEGIN

IMPORTS
    mail, mime-mhs, mime-mhs-bodies
    FROM MIME-MHS;

mime-mhs-bp-data OBJECT IDENTIFIER ::=
    { mime-mhs-bodies 1}

mime-mhs-bp-parameter OBJECT IDENTIFIER ::=
    { mime-mhs-bodies 2}

mime-generic-data OBJECT IDENTIFIER ::=
    { mime-mhs-bp-data 1}

mime-generic-parameters OBJECT IDENTIFIER ::=
    { mime-mhs-bp-parameter 1}

mime-postscript-body OBJECT IDENTIFIER ::=
    { mime-mhs-bp-data 2}

mime-jpeg-body OBJECT IDENTIFIER ::=
    { mime-mhs-bp-data 3}

mime-gif-body OBJECT IDENTIFIER ::=
    { mime-mhs-bp-data 4};
```

## 9. IANA Registration form for new mappings

To: IANA@isi.edu  
Subject: Registration of new X.400/MIME content type mapping

MIME type name:

(this must have been registered previously with IANA)

X.400 body part:

X.400 Object Identifier for Data:

(If left empty, an OID will be assigned by IANA under mime-mhs-bp-data)

X.400 Object Identifier for Parameters:

(If left empty, an OID will be assigned by IANA under mime-mhs-bp-parameter. If it is not used, fill in the words NOT USED.)

#### X.400 ASN.1 Syntax:

(must be an EXTENDED-BODY-PART-TYPE macro, or reference to a Basic body part type)

#### Conversion algorithm:

(must be defined completely enough for independent implementation. It may be defined by reference to RFCs).

Person & email address to contact for further information:

#### INFORMATION TO THE SUBMITTER:

The accepted registrations will be listed in the "Assigned Numbers" series of RFCs. The information in the registration form is freely distributable.

### 10. Security Considerations

Security issues are not discussed in this memo.

### 11. Authors' Addresses

Harald Tveit Alvestrand  
SINTEF DELAB  
N-7034 Trondheim  
NORWAY

EMail: Harald.Alvestrand@delab.sintef.no

Steven J. Thompson  
Soft\*Switch, Inc.  
640 Lee Road  
Wayne, PA 19087

Phone: (215) 640-7556  
EMail: sjt@gateway.ssw.com

## 12. References

- [1] Alvestrand, H., Kille, S., Miles, R., Rose, M., and S. Thompson, "Mapping between X.400 and RFC-822 Message Bodies", RFC 1495, SINTEF DELAB, ISODE Consortium, Soft\*Switch, Inc, Dover Beach Consulting, Inc., Soft\*Switch, Inc., August 1993.
- [2] CCITT Recommendation X.420 (1988), Interpersonal Messaging System.
- [3] Borenstein, N, and N. Freed, "MIME: Mechanisms for Specifying and Describing the Format of Internet Message Bodies", RFC 1341, Bellcore, Innosoft, June 1992.
- [4] ISO 8613; Information Processing: Text and Office System; Office Document Architecture (ODA) and Interchange Format (ODIF), Part 1-8, 1989.
- [5] ISO/IEC International Standard 10021, Information technology - Text Communication - Message-Oriented Text Interchange Systems (MOTIS) (Parts 1 to 8).
- [6] CCITT Recommendation T.411 (1988), Open Document Architecture (ODA) and Interchange Format, Introduction and General Principles.
- [7] Crocker, D., "Standard for the Format of ARPA Internet Text Messages", STD 11, RFC 822, UDEL, August 1982.
- [8] Hardcastle-Kille, S., "Mapping between X.400(1988) / ISO 10021 and RFC-822", RFC 1327, University College London, May 1992.
- [9] CCITT Recommendation T.4, Standardization of Group 3 Facsimile Apparatus for Document Transmission (1988).
- [10] CCITT Recommendation T.30, Procedures For Document Facsimile Transmission in the General Switched Telephone Network (1988).
- [11] CCITT, Data Communication Networks - Message Handling Systems - Recommendations X.400 - X.420 (1988 version).
- [12] Alvestrand, H., "X.400 Use of Extended Character Sets", RFC 1502, SINTEF DELAB, August 1993.