MIME: The Proposed Standard for Enhanced Internet Mail

Nathaniel S. Borenstein
Bellcore

Tutorial for CSCW ’92
Toronto, Ontario
November, 1992

MIME: The Proposed Standard for Enhanced Internet Mail

Agenda

Sunday, November 1, 1992

2:00-2:30: Introduction, Motivation, and Technical Background

2:30-3:00: MIME History, Design Rationale, and Technical Overview

3:00-3:30: The MIME Content-types for text messages

3:30-4:00: Break

4:00-4:30: The Non-text MIME Content-types

4:30-4:50: Minimal MIME, Extended MIME, and MIME Controversies

4:50-5:20: Adjuncts to MIME: Non-ASCII headers and the Metamail Program

5:20-5:30: MIME Status and Future Prospects

MIME: The Proposed Standard for Enhanced Internet Mail

Nathaniel S. Borenstein
Bellcore

Tutorial

Abstract

This tutorial presents a technical overview of MIME (Multipurpose Internet Mail Extensions), the new proposed standard format for Internet mail messages. MIME is a standard of great potential importance for CSCW, as it has the potential to vastly extend the functionality of electronic mail and its usefulness for CSCW applications. With MIME, email messages can include graphics, audio, and video in a standard format. This format can itself be extended to create new specialized message types, such as calendar queries, or even general-purpose messages that interact with their recipients and take differential actions based on those interactions, a capability of great interest in CSCW applications.

This tutorial will present both an overview of MIME's functionality and enough technical information to allow the designers of CSCW systems to build MIME-based applications. By the end of the tutorial, attendees should have a clear idea of what MIME can be used for and enough technical information to begin designing MIME-based applications almost immediately.

This tutorial will cover the philosophy behind MIME, the basic structure of email data in MIME format, the specific data types that are predefined by MIME, and the mechanisms and procedures for extending MIME to include new types of data. Additionally, the tutorial will outline two technologies that are complementary to MIME, a facility for including non-ASCII text in message headers, and a public domain MIME implementation that can be easily incorporated into various mail and other applications. All of this will be made concrete through the presentation of numerous examples of MIME in operation.

MIME: The Proposed Standard for Enhanced Internet Mail

Instructor Biography

Nathaniel S. Borenstein, a Member of Technical Staff at Bellcore in Morristown, New Jersey, is a research computer scientist specializing in human-computer interaction technologies and in electronic messaging, a field where he has been an active researcher for over a decade. While at Carnegie Mellon University, he was one of the primary authors of the Andrew Message System, a widely-used multimedia mail and bulletin board system. More recently, he has written metamail, a tool for making text-only mail reading software multimedia-capable, and ATOMICMAIL, a language for sending secure and portable interactive messages via electronic mail. He is the author of two books, the most recent of which is *Programming As If People Mattered: Friendly Programs, Software Engineering, and Other Noble Delusions*, from Princeton University Press. He is also one of the primary authors of MIME, the topic of this tutorial.

Table Of Contents

MIME: The Proposed Standard for Enhanced Internet Mail

Tutorial Outline

IX. Non-ASCII Characters In Message Headers

X. Bringing MIME to an Installed Base: The Metamail Software

XI. Status Report

XII. Implications for Email and Beyond

XIII. Access to MIME

I. Introduction and Motivation

Why Extend Email?

Text mail, fax, voice mail very popular.

Text mail deficient for non-English speakers due to ASCII character set.

Modern computer hardware can handle images and audio, too.

Integrated multimedia mail, as in Andrew, Slate, NeXT, etc., has wide appeal.

X.400, which purported to solve the problem, does not.

Why Is Data Interchange Hard?

Hundreds of Incompatible Formats

Standards have failed in two ways

ODA -- re-invents world, still not enough!

X.400 -- got distracted, re-invented the wrong things.

Heterogeneous formats seem here to stay.

What is MIME?

MIME: Multipurpose Internet Mail Extensions

Official proposed standard format for Internet mail messages.

Puublished as RFC 1341.

Approved as Proposed Standard by Internet Activities Board in June, 1992.

Basically MIME provides five things:

1. A way of labeling the type of data in non-text messages or non-ASCII text messages.

2. A way of delimiting where such data start and end in multipart messages

3. A mechanism for encoding arbitrary data for portable mail transport without information loss.

4. A set of initially defined "standard" data types.

5. A mechanism for defining and registering new data types.

To understand MIME's mechanisms, we must first understand the existing state of Internet mail.

II. Background: Email Today

X.400 vs.Internet mail

X.400: The top-down standard

Extremely complex

Relatively little-used in practice

Incomplete for multimedia

Requires complete software revolution from today’s world

Internet Mail: the *de facto* standard

Defined by RFC 821/822

Basis for most workstation & PC mail environments

Even more incomplete than X.400

Non-standard multimedia extensions (Andrew, Next, etc.)

The plan: Let email evolve gracefully with standardized Internet multimedia extensions.

First: need to understand the Internet mechanisms

The Mechanisms of Internet Mail

The Transport Layer

SMTP (Simple Mail Transfer Protocol, defined by RFC 821)

Extremely simple protocol for passing a message from one machine to the next.

Text-oriented protocol: limits on line length, only 7-bit data permitted.

This implies *raw binary data can not be transported by SMTP*.

There are "outlaw" SMTP implementations that support binary.

There have been proposals to redefine SMTP to permit binary transport.

These proposals have been rejected by the Internet community, for good reasons:

-- existing implementations will break

-- a raw binary transport channel does nothing to tell the recipient the type of data being transmitted

-- therefore extending SMTP involves much pain for little gain.

MIME was designed to require *no changes to SMTP*.

A parallel effort to extend SMTP continues, but in a MIME-compatible way.

The Internet Mail Data Format

The format of the messages transported by SMTP is defined by RFC 822.

This is an extremely simple, text-only format.

Each message has two parts, a header and a body, separated by the first pair of consecutive line breaks.

The body is plain ASCII (7-bit) text, of limited line length.

The header is a set of structured fields providing information about the message.

Each line is a different field, unless it begins with white space, which makes it a continuation line.

```
Field-1-Name:  Field-1-Body
From: Nathaniel Borenstein
   <nsb@bellcore.com>
To: Ned Freed
   <ned@innosoft.com>
Subject:  Hello There

This is a plain old 7-bit
ASCII text mail body.
```

MIME was designed to be backward-compatible with RFC 822:

> MIME messages should not break RFC 822 mail readers.
>
> MIME readers should handle older RFC 822 messages properly.

III. MIME: History and Design Rationale

Designed by Internet Engineering Task Force Working Group on Email Extensions

Politics: remarkable alliance between X.400-lovers and X.400-haters

> X.400-lovers see MIME as helping interconnect X.400 "islands"

Some arguments between SMTP-changers and SMTP-preservers.

> Resolution: MIME works with standard SMTP. A separate effort to extend SMTP, if successful, may make one small part of MIME (transfer encodings) unnecessary.

History

RFC 1049 defined Content-type header in March, 1988

Generalizations proposed at IFIP WG 6.5 conference in Zurich, fall 1990.

IETF (Internet Engineering Task Force) Working Group formed, Fall 1990.

First MIME draft, Spring 1991.

IAB (Internet Activities Board) approved as Proposed Internet Standard, Spring 1992.

Primary Design Goals of MIME

Text mail in any human language.

Non-text mail in any conceivable media type.

Complete compatibility with RFC 821 (SMTP) and RFC 822.

Total robustness over all known real-world email transport systems. (ASCII-EBCDIC BITNET gateways, for example)

Openness to multiple well-known formats.

Easy Extension to new types and formats.

Formal mechanism for type registration to promote interoperation.

Easy interoperation with X.400.

Compatibility with future SMTP extensions without dependence on them.

Ability for graceful, evolutionary introduction in the Internet.

IV. MIME: Technical Overview

MIME allows extended bodies, with type information in header fields.

"Content-Transfer-Encoding" field specifies data encoding algorithm

"Content-Type" provides type/subtype, optional parameters

"Content-Description" gives a textual description of the body data

"Content-ID" gives unique identifier for body parts, similar to Message-ID for messages.

"Content-" are the only header fields that matter for MIME body parts.

The Content-Type Header Field

Content-type header MUST specify type & subtype, separated by "/":

```
Content-type:   image/jpeg
```

Optional parameters are preceded by semicolons, in name=value form:

```
Content-type:
   application/ODA;
   profile=Q112
```

The default content-type is the one that is implicit in RFC 822:

```
Content-type: text/plain;
   charset=US-ASCII
```

MIME defines 7 main types, with most extension expected via subtypes.

New types should be reserved for wholly new media, such as smell, virtual reality, etc.

Primary type thus allows "reasonable" treatment of *unrecognized* subtypes:

text/*: show to user in raw form

Most others: Don’t.

multipart/*: show individual parts

The Content-Transfer-Encoding Header Field

Binary and long-line data does not survive email transport

Need to specify encoding algorithm for email transport

2 MIME transfer-encoding algorithms

Base64 algorithm is relatively dense, unreadable.

"Uuencode done right" (more robust)

Same algorithm used by PEM

Preferred encoding for binary data

Quoted-printable leaves most ASCII unchanged

- Encodes non-ASCII characters as "=A1"(hex)

- Preferred encoding for largely-ASCII data (e.g. French text)

NOTE: No encoding is permitted on content-types that include other content-types (message,multipart):

- Preserves structural transparency

- Nested encodings are wasteful

- Never really needed anyway

All encoding, therefore, takes place in "leaf" nodes of a structured multpart message.

The Base64 Encoding

The densest simple (non-compressed) encoding possible for email.

33% data expansion -- encodes 3 binary bytes in 4 ASCII characters by "moving byte boundaries"

Uses only A-Z a-z 0-9 + / =

Line breaks keep encoded lines short, ignored on decoding.

Two special cases:

> Line breaks always canonicalized to SMTP form (CR LF).
>
> "=" used for padding when total byte count is not a multiple of 3.

Base64 rearranges the 24 bits from 8 binary bytes into 4 6-bit "bytes" in a 64 character alphabet.

Consider the encoding for "ABC" (ASCII 65 66 67):

```
    A        B        C
01000001 01000010 01000011

010000 010100  001001 000011
  16     20      9      3
  Q      U       J      D
```

16, in base64 alphabet, is Q, 20 is U, 9 is J, and 3 is D, so "ABC" encodes as "QUJD"

The Quoted-Printable Encoding

Maximizes readability of included ASCII

All ASCII chars unchanged except "="

Other chars represented as "=0A", etc.

Long lines wrapped with "soft" line breaks, which are lines that end with "="

Special rules deal with meaning of white space at end of line

Up to 200% expansion.

Example (where "\255" means the byte with hex code FF):

"4=\255"   --->   "4=3D=FF"

Simple Examples

From: Nathaniel Borenstein
<nsb@thumper.bellcore.com>
To: Ned Freed <ned@innosoft.com
Content-type: text/plain
Content-Transfer-Encoding:
quoted-printable

This is text with a single non-ASCII character, =FF.

From: Nathaniel Borenstein
<nsb@thumper.bellcore.com>
To: Ned Freed <ned@innosoft.com
Content-type: image/gif
Content-Transfer-Encoding: base64

R0lGODdhSgGgAfUAAENDQ01NTTw8PEVF...

V. The MIME Content-types

MIME defines seven content-types. Most extension is expected via subtypes.

1. The "text" content-type

Subtypes: plain, richtext. (More expected.)

Critical parameter: charset.

"text/plain; charset=iso-8859-1" permits French email.

Other "official" character sets: ISO-8859-[1-9]. Permits mail in European languages, Hebrew, Arabic, and more.

Already in *de facto* use: "ISO-2022-jp" for Kanji text.

Deferred, but expected: Other Asian languages, UNICODE, ISO 10646. Politics...

text/richtext is an extremely simple "common denominator" markup language for enriched text.

Bold, italic, etc.

Indentation, centering, etc.

Excerpts, signatures

Future text subtypes: any largely-readable textual format.

A plain text message in US-ASCII:

From: Nathaniel Borenstein
<nsb@thumper.bellcore.com>
To: Ned Freed <ned@innosoft.com>
Subject: Plain text mail
Content-type: text/plain;
charset=us-ASCII

This is plain text mail.

A plain text message in French (using ISO-8859-1, quoted-printable encoding)

From: Nathaniel Borenstein
<nsb@thumper.bellcore.com>
To: Ned Freed <ned@innosoft.com>
Subject: French mail
Content-type: text/plain;
charset=iso-8859-1
Content-transfer-encoding:
quoted-printable

Le courrier =E9lectronique =E0 la fran=E7aise n=E9cessite quelques caract=E8res sp=E9ciaux pour faciliter la t=E2che du lecteur et =E9viter les ambigu=EFt=E9s

displays as....

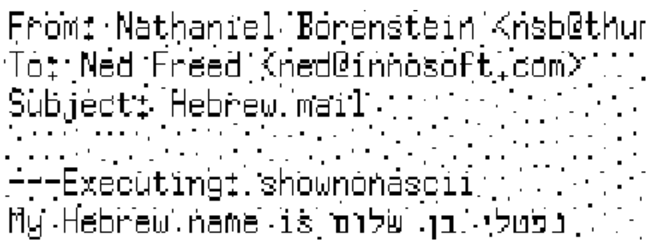
From: Nathaniel Borenstein
<nsb@thumper.bellcore.com>
To: Ned Freed <ned@innosoft.com>
Subject: French mail

Le courrier électronique à la française nécessite quelques caractères spéciaux pour faciliter la tâche du lecteur et éviter les ambiguïtés

A plain text message in Hebrew (using ISO-8859-8, quoted-printable encoding)

From: Nathaniel Borenstein
<nsb@thumper.bellcore.com>
To: Ned Freed <ned@innosoft.com>
Subject: Hebrew mail
Content-type: text/plain;
charset=iso-8859-8
Content-transfer-encoding:
quoted-printable

My Hebrew name is =ED=E5=EC=F9 =EF=E1 =E9=EC=E8=F4=F0



A Simple richtext message

From: Nathaniel Borenstein
<nsb@thumper.bellcore.com>
To: Ned Freed <ned@innosoft.com>
Subject: Richtext mail
Content-type: text/richtext;
charset=us-ASCII

This is <bold> enriched </bold> mail. Note the <italic> dramatic </italic> use of fonts.

...looks like this....

From: Nathaniel Borenstein
<nsb@thumper.bellcore.com>
To: Ned Freed <ned@innosoft.com>
Subject: Richtext mail

This is **enriched** mail. Note the *dramatic* use of fonts.

A Hebrew/English richtext message

From: Nathaniel Borenstein
<nsb@thumper.bellcore.com>
To: Ned Freed <ned@innosoft.com>
Subject: Hebrew richtext
Content-type: text/richtext;
charset=iso-8859-8
Content-transfer-encoding:
quoted-printable

This is <bold> enriched
=FA=E9=F8=E1=F2 </bold> mail.


From: Nathaniel Borenstein <nsb@thumper.bellcore.com>
To: Ned Freed <ned@innosoft.com>
Subject: Hebrew richtext

---Executing: shownonascii
This is **enriched עברית** mail.

2. The "image" content-type

Subtypes GIF, JPEG.

Widely-used, standard definitions

JPEG is more compact, GIF requires less processing to display.

Others subtypes expected, but not hoped for. Politics...

MTA Conversion possible, but a tricky business. See RFC 1344.

Base64 encoding preferred.

3. The "audio" content-type

Subtype "basic" for single-channel 8Khz u-law.

A compromise -- lowest common denominator audio

Other subtypes expected (compressed, high-fidelity, CD-quality, MIDI, etc.)

Again, MTA Conversion may be possible, but a tricky business. See RFC 1344.

Base64 encoding preferred.

4. The "video" content-type

Subtype "mpeg".

Others expected: H.261

Again, MTA Conversion may be possible, but a tricky business. See RFC 1344.

Data size is large enough to break almost all existing mail transport.

Recommended implementation technique: use message/external-body (see below)

5. The "multipart" content-type

Allows multiple body parts of different types, each structured like a mini-message.

Subtypes: mixed, alternative, parallel, digest.

Mixed: simple (serial) combinations.

Alternative: multiple representation of the same data.

Parallel: for parallel presentation *if possible*.

Digest: has special defaults for message digests (default content-type is message/rfc822 instead of text/plain)

All multipart subtypes share a common syntax.

Allows future experimentation in structured types, etc.

Multipart/foobar may be unrecognized, but all MIME readers can find & try to display the parts.

Future multipart subtypes might provide organized interactive presentations.

Multipart: The Common Syntax

The multipart content-type field must include a "boundary" parameter, e.g.:

Content-type: multipart/mixed;
    boundary=foobar

Within the body of the message, each body part is prefixed by a line consisting of "--" and the boundary.

After last part comes the "--" line, the boundary, and "--".

Anything before the first boundary line or after the last boundary line is ignored by MIME readers.

The "prefix" area before the first boundary may be used to alert non-MIME mail readers to what’s up.

From: Nathaniel Borenstein
<nsb@bellcore.com>
To: Ned Freed <ned@innosoft.com>
Subject: A multipart example
Content-Type: multipart/mixed;
boundary=CUT_HERE

WARNING: This is a message in MIME format. If you're reading this warning, your mail reader does not understand MIME, and you may want to look into upgrading it.
--CUT_HERE
Content-type: text/plain

Hey, Ned, look at this neat picture:
--CUT_HERE
Content-type: image/gif
Content-Transfer-Encoding: base64

5WVlZ6enqqqr....

```
--CUT_HERE
Content-type: text/richtext;
    charset=iso-8859-8
Content-transfer-encoding:
    quoted-printable

Wasn't that <bold>neat?</bold>
    --=ED=E5=EC=F9 =EF=E1
=E9=EC=E8=F4=F0
--CUT_HERE--
(This area is ignored by MIME readers.)
```

```
From: Nathaniel Borenstein <nsb>
To: Ned Freed <ned@innosoft.com>
Subject: An alternative example
Content-Type: multipart/alternative;
    boundary=CUT_HERE

--CUT_HERE
Content-type: text/plain

Hey, Ned, Isn’t MIME great?
--CUT_HERE
Content-type: text/richtext

Hey, <bold> Ned</bold>, isn’t
MIME <italic>great</italic>?
--CUT_HERE
Content-type: application/ODA
Content-Transfer-Encoding: base64

5WVlZ6enqqqr....
--CUT_HERE--
```

```
From: Nathaniel Borenstein <nsb>
To: Ned Freed <ned@innosoft.com>
Subject: A digest example
Content-Type: multipart/digest;
     boundary="------"

--------

From: Sender1
Subject: First digested  message

blah blah blah
--------

From: Sender2
Subject: Second digested  message

blah blah blah
----------
```

6. The "message" content-type

Subtypes: rfc822, partial, external-body

"message/rfc822": encapsulated message.

"message/partial": for automatic fragmentation and reassembly.

"message/external-body" allows data to be passed by reference (hyperlinks).

Supported access methods: local-file, afs, ftp, tftp, anon-ftp, mail-server

```
From: nsb@thumper.bellcore.com
To: Ned Freed <ned@innosoft.com>
Subject: A forwarded message
Content-type:  multipart/mixed;
     boundary=foobar

--foobar

Here's  a message  I wanted to forward
to you.  Note that  MIME formatting is
preserved easily and recursively.
--foobar
Content-Type: message/rfc822

From: someone else
Content-type: image/gif
Content-Transfer-Encoding: base64

QSDLKJEWRUIOsdfkjao.......
--foobar--
```

```
From: Nathaniel Borenstein
    <nsb@thumper.bellcore.com>
To: Ned Freed <ned@innosoft.com>
Subject: A partial example
Content-Type: message/partial;
    number=1; total=2;
    id="unique-id"

Content-type: image/gif
Content-transfer-encoding: base64

5WVlZ6enqqqr...
----------------
From: Nathaniel Borenstein
    <nsb@thumper.bellcore.com>
To: Ned Freed <ned@innosoft.com>
Subject: A partial example
Content-Type: message/partial;
    number=2; total=2;
    id="unique-id"

Ozs3h4eIKCgo6Ojp...
```

```
From: Nathaniel Borenstein <nsb>
To: Ned Freed <ned@innosoft.com>
Subject: Some external references
Content-Type: multipart/alternative;
    boundary=42

--42
Content-Type: message/external-body;
    access-type=mail-server
    server="listserv@bogus.bitnet"

Content-type: application/postscript

get rfc-xxxx doc
--42
```

```
Content-Type: message/external-body;
     name="BodyFormats.ps";
     site="thumper.bellcore.com";
     access-type=ANON-FTP;
     directory="pub/nsb"

Content-type: application/postscript
--42
Content-Type: message/external-body;
     name="/u/nsb/BodyFormats.ps";
     site="thumper.bellcore.com";
     access-type=local-file

Content-type: application/postscript
--42--
```

7. The "application" content-type

Catch-all.

Subtypes: PostScript, ODA.

Most creative extensions expected here.

Possible examples:

- EDI
- Acknowledgement-request
- Interactive-survey
- WWW, WAIS, Gopher, ...?

Already registered:

- ATOMICMAIL, Andrew-inset

A Complex MIME Message

From: Nathaniel Borenstein
<nsb@thumper.bellcore.com>
To: Ned Freed <ned@innosoft.com>
Subject: A complex example
Content-Type: multipart/mixed;
boundary=FOOBAR

This is the preamble area, which is ignored.
--FOOBAR
Content-type: text/plain;
charset=US-ASCII

PLAIN TEXT GOES HERE.
--FOOBAR
Content-Type: audio/basic
Content-Transfer-Encoding: base64

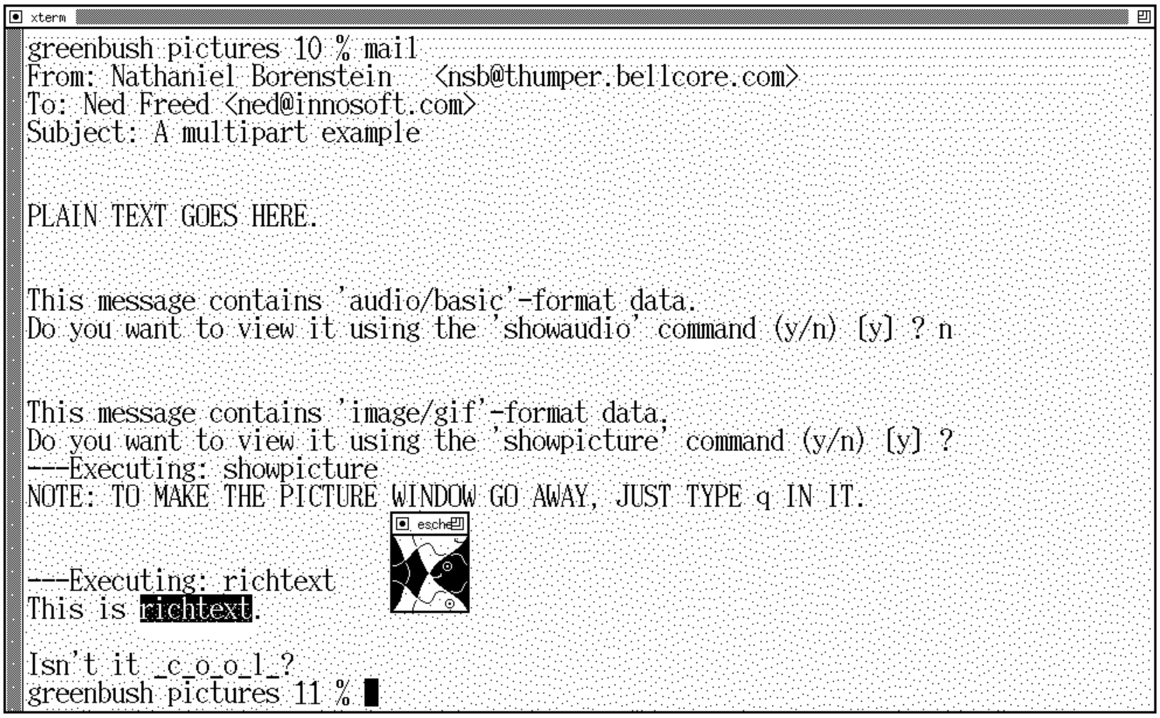
BASE64-ENCODED AUDIO DATA

--FOOBAR
Content-Type: image/gif
Content-Transfer-Encoding: Base64

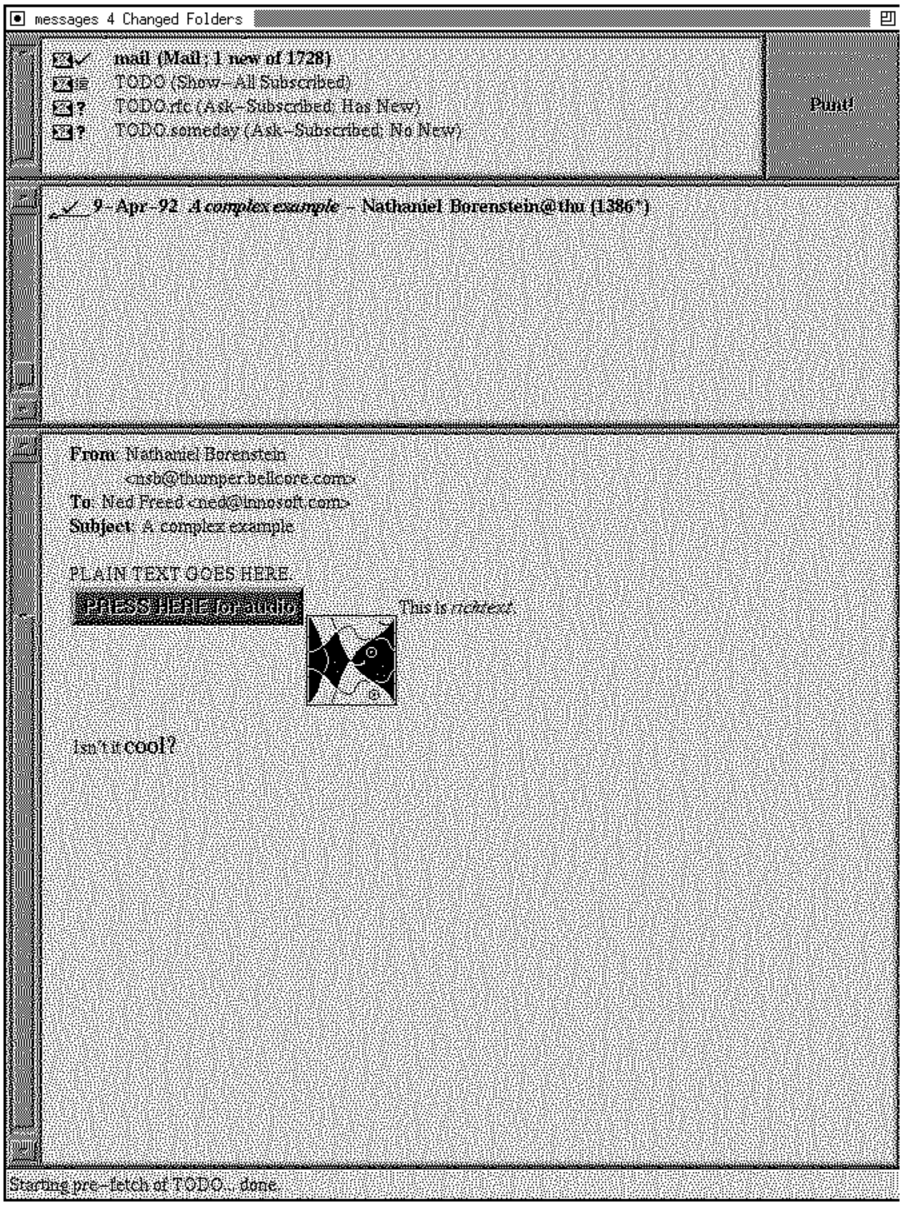
BASE64-ENCODED IMAGE DATA GOES HERE
--FOOBAR
Content-type: text/richtext

This is <italic>richtext</italic>.
<nl><nl> Isn’t it <bigger>cool?</bigger>
--FOOBAR--
This is the "epilogue" area, also ignored.

What the Complex Example Looks like with a standard mail reader



What the Complex Example Looks like with an integrated multimedia mail reader



VI. Minimal MIME-Conformance

MIME is open-ended, needs interoperable core

"MIME-conformance" defined to require:

1. "MIME-Version: 1.0"

2. Content-Transfer-Encoding

3. text/plain; charset=US-ASCII

4. ASCII portions of ISO-8859-*

5. Intelligent treatment of unrecognized types & charsets

6. message/rfc822

7. multipart (mixed, alternative, *)

VII. Extension Mechanisms

A major goal of MIME is to let you send ANYTHING through the mail.

Obviously, the MIME standard can't enumerate all possible data types.

But there needs to be a way to ensure consistency without overly constraining innovation:

--All systems sending a data type should use the same content-type values for it.

--Two systems should not use the same name for two different data types

--Informal experimentation should still be possible.

Informal experiments:

Anyone can create a subtype (or type) name starting with "X-"; these are considered experimental.

Similarly, "X-" is allowed for character sets, external-body access-types, etc.

Allows independent development of new ideas for email applications

Formal extensions:

- IANA (Internet Assigned Names Authority) keeps a registry of MIME values (content-types, etc.).
- Names not starting with "X-" must be registered with IANA.
- For IANA registration, a stable and public specification must be published.

Formal extensions are *not* mandatory for MIME-conformant implementations.

- To be mandatory, they have to pass through the Internet standards process.
- Such mandates are rarely likely to be necessary.
- Even most content-types in the MIME definition are not considered mandatory!

MIME as a platform for CSCW Experiments

Facilitates new mail-based applications.

MIME extension mechanisms allow flexible experimentation.

Example: Interactive messaging

-- Labelled by MIME type "application/atomicmail"

-- In multipart, can exploit other MIME types.

-- Support is easily installed in flexible MIME readers.

MIME gives a path to standardizing the useful applications.

VIII. Controversies and Problems

MIME was the work of 100 Internet mail gurus. There were arguments.

Nested encodings were banned.

Compression was punted.

Uuencode was rejected.

A proliferation of top-level types was rejected.

X.400 interoperation affected many things:

> No use of preamble/epilogue in multipart type.
>
> Parts are NOT encapsulated messages.

More Controversies and Problems

Part boundaries do not depend on line or byte counts.

Binary or 8-bit transport neither presumed nor prohibited.

"Preferred" encodings are OK, but no requirements.

Non-ASCII header texts specified in companion memo.

Initial character sets limited to ISO-8859 family.

IX. Non-ASCII Characters In Message Headers

MIME, as defined by RFC 1341, only addresses message bodies.

The need for non-ASCII header text is addressed by RFC 1342, a companion document to MIME.

Rationale:

More complex

More ugly

Less consensus

Non-ASCII header data is handled as a special case.

- Only text (no images, etc.) is permitted
- Non-ASCII text is permitted only in certain very special locations
- All of this is complicated and made uglier by RFC 822 header syntax.
- Old parsers of RFC 822 headers should not fail with these extensions.

Where Non-ASCII Data Are Permitted

Only in fields intended for human reading (not automatic processing):

Subject field

Subject: **NON-ASCII**

Comments field

Comments: **NON-ASCII**

Syntactic comments in any field

From: nsb (**NON-ASCII**)

Route phrase in email addresses

From: **NON-ASCII** <x@y>

How Non-ASCII Data Are Represented

An ugly scheme permits "encoded words" to be inserted in any of the permitted locations.

Encoded words must be short; long ones may be broken into multiple encoded words, to avoid line length problems.

No line breaks within encoded words

Uses standard MIME encodings (base64, quoted-printable)

Anything that includes key syntactic characters (e.g. "()?=_<>",and more) must be encoded

All spaces are mapped to underscores

Special syntax delimits start & end of encoded string, specifies charset & encoding

General form:

=?CHARSET?ENCODING?DATA?=

Charset values same as MIME

Encoding values:

B for base64

Q for quoted-printable

Example of Non-ASCII Header Data

Here is an example of email containing human names in Hebrew and Japanese:

From: Nathaniel Borenstein <nsb@thumper.bellcore.com> (=?ISO-8859-8?Q?=ED=E5=EC=F9 =EF=E1 =E9=EC=E8=F4=F0?=)
To: Yutaka Sato <ysato@etl.go.jp> (=?ISO-2022-JP?B?GyRAOjRGI0stGyhK?=)


From: Nathaniel Borenstein <nsb@thumper.bellcore.com> (נפטלי בן שלום)
To: Yutaka Sato <ysato@etl.go.jp> (佐藤豊)
Subject: International mail

X. Bringing MIME to an Installed Base: The Metamail Software

MIME solves the data format problem, but doesn't itself address the problem of the installed base.

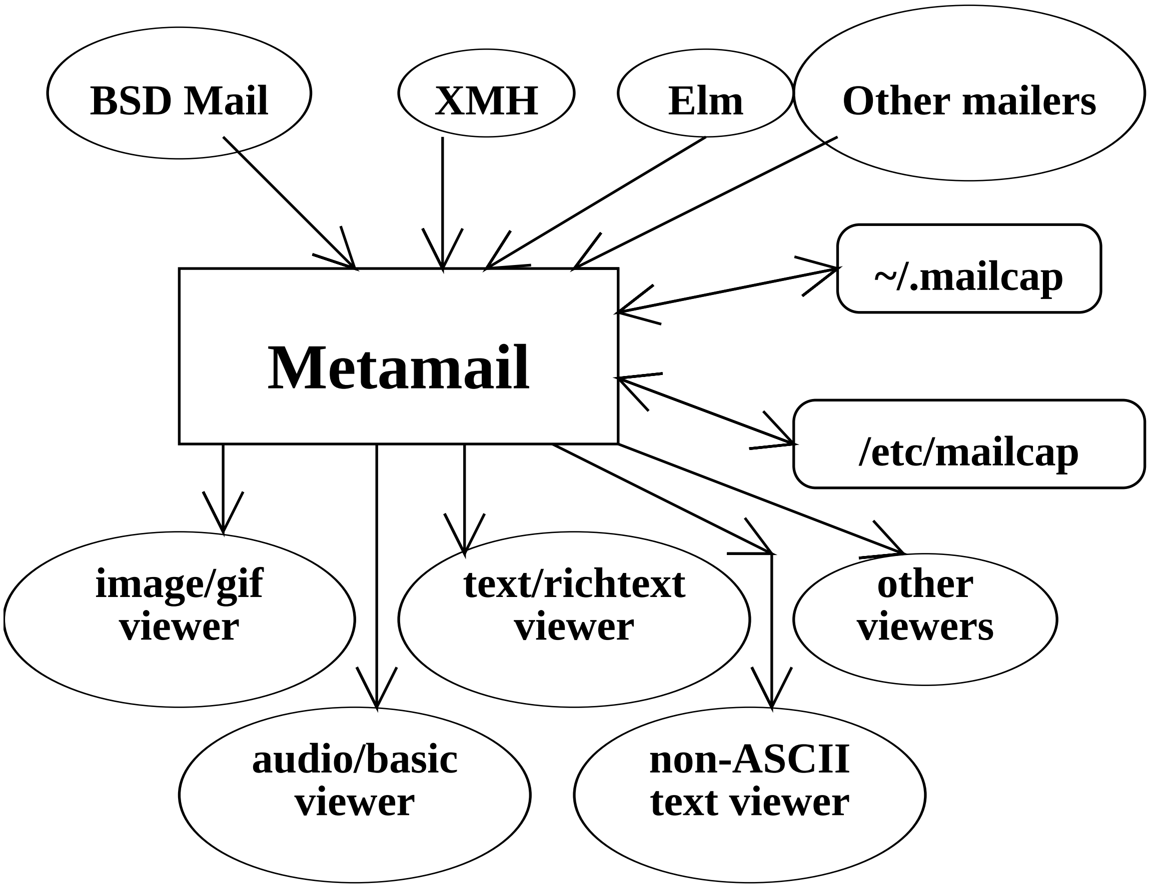
People won't send multimedia mail regularly until most people can read it.

People are understandably slow and reluctant to change mail reading tools.

Needed: an easy way to upgrade existing text-only mail readers to be MIME-conformant.

This is the role played by the "metamail" software, which can be used as the glue that binds older software to MIME.

The Metamail Architecture



Life with Metamail

Support for new types of mail can be easily added to *all* patched mail readers with a single line in a mailcap file.

A metamail-patched mail reader is *not* an integrated multimedia mail system!

However, it is a low-pain way to gain the capability to read MIME mail.

Users who get lots of MIME mail will eventually be motivated to switch to an integrated MIME reader, but now that's entirely up to them.

The main goal of metamail is to facilitate the transition to a multimedia mail world.

What Metamail Includes

Core metamail program is just a big switch, connecting mail readers to media viewers.

> Understands non-ASCII header text.
>
> Built-in: multipart/mixed, multipart/alternative

Metamaila runs on all UNIX variants, plus MS-DOS and Commodore Amiga.

The metamail software can be patched into existing applications, or its code can be cannibalized in building integrated MIME applications.

Metamail distribution includes simple viewers for nearly all predefined MIME formats.

All work on UNIX, many work on other systems.

- Terminal-oriented richtext viewer
- Image and audio viewers
- message/partial and message/external-body
- X11 Viewers for non-ASCII text.

Metamail distribution also includes:

- mailto, a simple terminal-oriented MIME mail composer.

- metasend, a script for packing up files of various types as MIME mail.

- splitmail, for breaking messages into message/partial pieces

- mmencode, a program for encoding and decoding base64 and quoted-printable data.

- Various contributed software, including a metamail server, DOS and Amiga add-ons, and more.

Metamail Installation

Installing metamail consists of two steps:

1. Compiling and installing the metamail program set
2. Patching the mail reading programs at your site.

Step 1 is automatic and completes in minutes

Step 2 requires that you have sources for your mail readers. Metamail comes with patch instructions for dozens of mail and news readers, and most patches are under 50 lines of code.

New releases of various mail readers will include the metamail patch, making step 2 increasingly unnecessary.

XI. Status Report

At least two dozen MIME implementations under way.

At least 4 public domain implementations available now.

Metamail is in daily use at hundreds of sites, incorporated into new release of several mail readers.

MIME -> Proposed Internet Standard

In winter 1992-3, The Internet Activities Board will probably be asked to advance MIME to "Draft Standard" status.

XII. Implications for Email and Other Multimedia Applications

Whither X.400? Choose one:

MIME is the death of X.400

MIME will help X.400 a lot.

Whither Fax?

Probably no effect.

Might facilitate future merging with email

Whither voice mail?

May help with interoperation.

MIME in non-mail applications

Paranoid email robustness can't hurt elsewhere.

It's nice to know your data format is mind-bogglingly robust.

Open architecture invites re-use.

MIME & metamail already used in non-email applications. (Superbook, INRIA database, experimental systems)

Interest in MIME/metamail for Gopher, WAIS, etc.

De Facto standards happen from the bottom up.

Future problems in multimedia data interchange

Constraining and defining the set of formats in use.

Standard formats for tight coupling of separate objects.

Standardized control structures for interactive applications.

New media types (e.g. smell, virtual worlds)

All of this seems to be MIME-able.

XIII. Access to MIME

There are four MIME-related RFCs:

RFC 1341 -- MIME
RFC 1342 -- Non-ASCII Headers
RFC 1343 -- Mailcap file format
RFC 1344 -- Implications of MIME for Gateways

To get RFCs:

anonymous ftp (password "guest") to FTP.NISC.SRI.COM, directory "rfc", file name "rfcnnnn.txt" or "rfcnnnn.ps", index file "rfc-index.txt"

Mail to MAIL-SERVER@NISC.SRI.COM with body "send rfcNNNN" or "send rfcNNNN.ps" or "send rfc-index"

For metamail info, send email:

To:
mailserver@thumper.bellcore.com
Subject: metamail-info

This will tell you how to get the code, among other things.

For more information:

Nathaniel S. Borenstein
<nsb@thumper.bellcore.com>

Bellcore
Bell Communications Research
nsB