MIME Overview

by Mark Grand <mark@premenos.sf.ca.us>

Internet e-mail allows mail messages to be exchanged between users of computers around the world and occasionally beyond… to space shuttles. One of the main reasons that Internet e-mail has achieved such wide use is because it provides a standard mechanism for messages to be exchanged between over 1,000,000 computers connected to the Internet.

The standards that are the basis for Internet e-mail were established in 1982. Though they were state of the art in 1982, in the intervening years they have begun to show their age. The 1982 standards allow for mail messages that contain a single human readable message with the restrictions that:

- the message contains only ASCII characters.
- the message contains no lines longer than 1000 characters.
- the message does not exceed a certain length

The 1982 standards do not allow EDI to be transmitted through Internet mail, since EDI messages can violate all of these restrictions. There are a number of other types of messages and services that have are supported by other mail standards that have been designed more recently. In June of 1992 a new Internet mail standard was approved. This new standard is called MIME.

MIME is an acronym for Multipurpose Internet Mail Extensions. It builds on the older standard by standardizing additional fields for mail message headers that describe new types of content and organization for messages.

MIME allows mail messages to contain:

- Multiple objects in a single message.
- Text having unlimited line length or overall length.
- Character sets other than ASCII.
- Multi-font messages.
- Binary or application specific files.
- Images, Audio, Video and multi-media messages.

MIME defines the following new header fields:

1. A `MIME-Version` header field, which uses a version number to declare that a message conforms to the MIME standard.

2. A `Content-Type` header field, which can be used to specify the type and subtype of data in the body of a message and to fully specify the encoding of such data.

    2.a. A `Text Content-Type` value, which can be used to represent textual information in a number of character sets and formatted text description languages in a standardized manner.

    2.b. A `Multipart Content-Type` value, which can be used to combine several body parts, possibly of differing types of data, into a single message.

    2.c. An `Application Content-Type` value, which can be used to transmit application data or binary data.

    2.d. A `Message Content-Type` value, for encapsulating a mail message.

    2.e. An `Image Content-Type` value, for transmitting still image (picture) data.

    2.f. An `Audio Content-Type` value, for transmitting audio or voice data.

    2.g. A `Video Content-Type` value, for transmitting video or moving image data, possibly with audio as part of the composite video data format.

3. A `Content-Transfer-Encoding` header field, that specifies how the data is encoded to allow it to pass through mail transports having data or character set limitations.

4. Two optional header fields that can be used to further describe the data in a message body, the `Content-ID` and `Content-Description` header fields.

MIME is an extensible mechanism. It is expected that the set of content-type/subtype pairs and their associated parameters will grow with time. Several other MIME fields, such as character set names, are likely to have new values defined over time. To ensure that the set of such values develops in an orderly, and public manner, MIME defines a registration process which uses the Internet Assigned Numbers Authority (IANA) as a central registry for such values.

To promote interoperability between implementations, the MIME standard document specifies a minimal subset of the above mechanisms that are required for an implementation to claim to conform to the MIME standard.

MIME Technical Summary

MIME is defined by an Internet standard document called RFC 1341. This document summarizes the contents of RFC 1341. Sufficient detail is presented here to understand the capabilities of MIME. For sufficient detail to implement MIME please read RFC 1341.

MIME allows messages to contain multiple objects. When multiple objects are in a MIME message, they are represented in a form called a *body part*. A body part has a header and a body, so it makes sense to speak about the body of a body part. Also, body parts can be nested in bodies that contain one or multiple body parts.

The `Content-Type` values, subtypes, and parameter names defined in the MIME standard are not case insensitive. However, many parameter values are case sensitive

The MIME standard is written to allow MIME to be extended in certain ways, without having to revise the standard. MIME specifies sets of values that are allowed for various fields and parameters. The provides a procedure for extending these sets of values by registering them with an entity called the Internet Assigned Numbers Authority (IANA).

The MIME-Version Header Field

MIME is designed to be compatible with older Internet mail standards. In particular, it is compatible with RFC 822. If a mail reading program receives a message that is a MIME message then it will likely perform additional processing for the MIME message that it would not perform for non-MIME messages. In order to allow mail reading programs to recognize MIME messages, MIME messages are required to contain a `MIME-Version` header field. The `MIME-Version` header field specifies the version of the MIME standard that the message conforms to.

As of this writing there is only version (1.0) of the MIME standard. Messages that comply with the standard must include a header field, with the following verbatim text:

```
MIME-Version: 1.0
```

The `MIME-Version` header field is required at the top level of a message. It is not required for each body part of a multipart entity. It is required for the embedded headers of a body of type "message" if and only if the embedded message is claimed to be MIME-compliant.

The Content-Type Header Field

The `Content-Type` field describes the data contained in the body fully enough that the mail reader can pick an appropriate mechanism to present the data to the user, or otherwise deal with the data in an appropriate manner.

The `Content-Type` header field is used to specify the nature of data in the body or body part, by giving type and subtype identifiers, and by providing parameters that may be

needed for certain types. After the type and subtype names, the remainder of the header field is a set of parameters, specified in an attribute/value notation. The set of meaningful parameters differs for different types. The order of parameters is not significant. Comments are allowed (in accordance with RFC 822 rules) in structured header fields by placing them in parentheses.

The top-level `Content-Type` is used to declare the general type of data, while the subtype specifies a specific format for that type of data. Thus, a `Content-Type` of `Image/xyz` is enough to tell a mail reader that the data is an image, even if the mail reader has no knowledge of the specific image format `xyz`. Such information can be used, to decide whether or not to show a user the raw data from an unrecognized subtype — such an action might be reasonable for unrecognized subtypes of `Text`, but not for unrecognized subtypes of `Image` or `Audio`. For this reason, registered subtypes of `Audio`, `Image`, `Text`, and `Video`, should not contain embedded information that is really of a different type. Such compound types are usually represented using the `Multipart` or `Application` types.

Parameters are modifiers of the content-subtype. Although most parameters make sense only with certain content-types, others are "global" in the sense that they might apply to any subtype. For example, the `Boundary` parameter, which is used to indicate how body parts are separated from each other, makes sense only for the `Multipart` content-type. The `Charset` parameter might make sense with several content-types.

The MIME standard defines seven content-types. The authors of the MIME standard state that the set of seven types is "substantially complete". They expect additional supported types to be accommodated by creating new subtypes of the seven initial top-level types. The MIME standard, functioning as a constitution for the MIME community, states that new standard content types can be defined only by revising the standard (as opposed to the registration procedure for other types of extensions). However, MIME does provide for the use of non-standard content types. Non-standard content-types can be used, but must be given names starting with `X-`. Future standard content type names will not begin with `X-`.

The syntax for the content type header field is

```
Content-Type := type "/" subtype [";" parameter]…
```

The defined content types are:

`Application`
: indicates data that does not fit into any of the other categories, such as uninterpreted binary data or information to be processed by a mail-based application. In addition to the following subtypes, it is likely that additional subtypes will be defined for applications such as mail-based scheduling systems, spreadsheets and EDI.

`Application/Octet-Stream`

indicates uninterpreted binary data, which a mail reading program may simply offer to write the information into a file. Possible parameters for `Application/Octet-Stream` include:

`Name`

a suggested name for the binary data if stored as a file.

`Type`

the general type or category of binary data. This is intended for human recipients rather than for automated processing.

`Conversions`

the operations that performed on the data before putting it the body. Note that the standard defines **no** conversion values. Any conversion values that do not begin with `X-` must be preceded by a published specification and by registration with IANA.

`Padding`

the number of bits of padding that were appended to the bitstream comprising the actual contents to produce the enclosed byte-oriented data. This is useful for enclosing a bitstream in a body when the total number of bits is not a multiple of the byte size.

`Application/ODA`

indicates a body containing information encoded according to the Office Document Architecture (ODA) 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 a `Profile` parameter. Thus an appropriate header field might look like this:

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

Consult the ODA standard for further information.

`Application/PostScript`

indicates a body containing a postscript document.

`Audio`

Indicates audio data. Audio requires an audio output device (such as a speaker or a telephone) to "display" the contents.

`Audio/Basic`

The content of the `Audio/Basic` subtype is audio encoded using 8-bit ISDN u-law. When this subtype is present, a sample rate of 8000 Hz and a single channel is assumed.

`Image`

Image data. Image requires a display device (such as a graphical display, a printer, or a FAX machine) to view the information.

`Image/Jpeg`

indicates an image in JPEG format.

`Image/Gif`

indicates an image in GIF format.

`Message`

indicates an encapsulated message.

`Message/Rfc822`

indicates that the body contains an encapsulated message, with the syntax of an RFC 822 message.

`Message/Partial`

indicates a partial message, allowing fragmented transmission of bodies too large to be passed through mail transport facilities. `Message/Partial` indicates that the body contains a fragment of a larger message.

Three parameters are required in a `Content-Type` field of type `Message/Partial`: The first, `Id`, is a unique identifier, as close to world-unique as possible, used to match the parts together. The second, `Number`, an integer, is the part number indicating where this part fits into the sequence of fragments. The third, `Total`, another integer, is the total number of parts. `Total` is required on the final part, and optional on earlier parts.

`Message/External-Body`

indicates that the actual body data are not included, but merely referenced. In this case, the parameters describe a mechanism for accessing the external data.

When a body or body part is of type `Message/External-Body`, it consists of a header, a blank line, and the message header for the encapsulated message. If another blank line appears, this ends the message header for the encapsulated message. However, since the encapsulated message's body is itself external, it does **not** appear in the area that follows. For example, consider this message:

```
Content-type: message/external-body;
              access-type=local-file;
              name=/u/nsb/Me.gif

Content-type:  image/gif

THIS IS NOT REALLY THE BODY!
```

The area at the end, which constitutes a *phantom body*, is ignored for most external-body messages. However, it may be used to contain auxiliary information for a "mail-server".

The only parameter of `Message/External-Body` that is always mandatory is `Access-Type`. Its other parameters are mandatory or optional depending on the value of `Access-Type`. The values defined for the `Access-Type` parameter are `FTP`, `ANON-FTP`, `TFTP`, `AFS`, `LOCAL-FILE`, and `MAIL-SERVER`. Except for values beginning with `X-`, other values must be registered with IANA.

The standard also specifies additional parameters that are to be used in conjunction with the various access types.

In addition to access-type specific parameters, the standard defines the following parameters which are optional for **all** access types:

- The `Expiration` parameter is used to specify a date after which the existence of the external data is not guaranteed.
- The `Size` parameter is used to specify the size of the data.

`Multipart`

indicates data consisting of multiple body parts; each having its own data type. It is possible to tell where each body part begins and ends because each body part is preceded by a special string called an *encapsulation boundary*; the last body part is followed by a *closing boundary*.

The boundary strings used are specified by a mandatory parameter called `Boundary`. The encapsulation boundary is an end of line followed by two hyphens followed by the boundary parameter value of the `Content-Type` header field. The closing boundary is the same as the encapsulation boundary with the addition of two hyphens at the end of the line.

The encapsulation boundary **must not** appear inside any of the encapsulated parts. It is crucial that the composing user agent be able to choose and specify the unique boundary that will separate the body parts. Encapsulation boundaries may be no longer than 70 characters, not counting the blank line and leading hyphens.

Thus, a typical multipart Content-Type header field might look like:

```
Content-Type: multipart/mixed; boundary=gc0y0pkb9ex
```

This indicates a body consisting of several body parts, each having a structure syntactically identical to an RFC 822 message, except that the header area may be completely empty, and each part is preceded by the line

```
--gc0y0pkb9ex
```

The closing boundary following the last body part indicates that no further body parts will follow. It is identical to the preceding encapsulation boundaries, with the addition of two more hyphens at the end of the line:

```
--gc0y0pkb9ex--
```

There is room for additional information prior to the first encapsulation boundary and following the final boundary. These areas are often blank. Anything appearing before the first or after the last boundary is ignored.

As a simple example, the following multipart message has two parts, both plain text, one explicitly typed and one implicitly typed:

```
From: Nathaniel Borenstein <nsb@bellcore.com>
To: Ned Freed <ned@innosoft.com>
Subject: Sample message
MIME-Version: 1.0
Content-type: multipart/mixed;
              boundary="simple boundary"

This is the preamble.  It is to be ignored, though it is
a handy place for mail composers to include an
explanatory note to non-MIME compliant readers.
--simple boundary


This is implicitly typed plain ASCII text.
--simple boundary
Content-type: text/plain; charset=us-ascii

This is explicitly typed plain ASCII text.
It DOES end with a line break.

--simple boundary--
This is the epilogue.  It is also to be ignored.
```

The use of a `Content-Type` of `multipart` in a body part within another multipart entity is explicitly allowed. In such cases, care must be taken to ensure that each nested multipart entity uses a different boundary delimiter.

The use of the `multipart` `Content-Type` with only a single body part may be useful in certain contexts, and is explicitly permitted.

`Multipart/Mixed`
: indicates multiple independent body parts to be viewed serially.

`Multipart/Alternative`
: is syntactically identical to `Multipart/Mixed`. Each part is an "alternative" version of the same information. Mail readers should recognize that the content of the parts are interchangeable. The mail reader should either choose the "best" type based on the user's environment and preferences, or offer the user the available alternatives. Generally, choosing the best type means displaying only the **last** part that can be displayed. This may be used, for example, to send mail in a fancy text format in such a way that it can easily be displayed anywhere:

  ```
  From:  Nathaniel Borenstein <nsb@bellcore.com>
  To: Ned Freed <ned@innosoft.com>
  Subject: Formatted text mail
  ```

```
MIME-Version: 1.0
Content-Type: multipart/alternative;
              boundary=boundary42


--boundary42
Content-Type: text/plain; charset=us-ascii

...plain text version of message goes here....
--boundary42
Content-Type: text/richtext

.... richtext version of same message goes here ...
--boundary42
Content-Type: text/x-whatever

.... fanciest formatted version of same  message
goes  here
...
--boundary42--
```

In this example, users whose mail system understood the `text/x-whatever` format would see only the fancy version, while other users would see only the richtext or plain text version, depending on the capabilities of their system.

Some mail reading programs that recognize more than one of the formats will offer the user a choice of which format to view. This makes sense, for example, if mail includes both a nicely formatted image version and an easily edited text version. The point is that multiple versions of the same data are not automatically shown. Either the user is shown the last recognized version or explicitly given the choice.

`Multipart/Parallel`
: is syntactically identical to `Multipart/Mixed`. However, in a parallel body, all of the body parts are intended to be presented simultaneously on hardware and software that are capable of doing so. Composing agents should be aware that many mail readers will lack this capability and will show the parts serially in any event.

  `Multipart/Parallel` will likely be used for multimedia messages that combine such message types as text, audio and/or video.

`Multipart/Digest`
: Indicates that each of the body parts is an RFC 822 mail message. `Multipart/Digest` is syntactically identical to `Multipart/Mixed`, except that the default `Content-Type` value for a body part is changed from `Text/Plain` to `Message/Rfc822`.

`Text`
: The `text Content-Type` is for sending material that is principally textual in form. It is the default `Content-Type`. A `Charset` parameter may be

used to indicate the character set of the text. The default `Content-Type` for Internet mail is `text/plain; Charset=US-ASCII`.

The value of the `Charset` parameter is not case sensitive. Allowable values are `US-ASCII`, `ISO-8859-1`, `ISO-8859-2`, ... and `ISO-8859-9`. The default value for `Charset` is `US-ASCII`.

`Text/Plain`
: indicates plain (unformatted) text. No special software is required to get the full meaning of the text, aside from support for the indicated character set. Other subtypes should be used for enriched text in forms where application software may enhance the appearance of the text, but such software must not be required in order to get the general idea of the content. Possible future subtypes include any readable word processor format.

`Text/Richtext`
: indicates a simple portable word processing format that is defined by the MIME standard. It is a very small subset of SGML. Mail readers that implement `Richtext` may implement only a subset of it.

  When a mail composing program is given a file in a word processing format to send and there is no standardized subtype for that format, then the message composing program may reformat the file into richtext format which will preserve more of the original formatting information than reformatting the file to plain ASCII.

`Video`
: indicates that the body contains a time-varying-picture image, possibly with color and coordinated sound. The term `Video` is used very generically and does not refer to any particular technology or format. It is not meant to preclude subtypes such as animated drawings encoded compactly.

  `Video/Mpeg`
  : indicates video coded according to the MPEG standard.

`X-`*`TypeName`*
: This is any type name that begins with `X-`. A `Content-Type` value beginning with `X-` is a private value, to be used by consenting mail systems by mutual agreement. The standard specifies no subtypes.

No type may be specified without a subtype.

The standard allows the use of additional sub-types without having to change the standard. However, it is important to insure that sub-types used by different user communities of MIME do not conflict. It would be confusing if `Content-Type: application/foobar` meant two different things. The standard specifies two mechanisms for defining new `Content-Type` subtypes:

1. Private values (starting with X-) may be defined between cooperating mail composing and reading programs without outside registration. Use of this mechanism requires knowing that the reader of the message will not mistake the content type for something other than originally intended.

2. New standard values must be registered with IANA. Where intended for public use, the formats they refer to must also be defined by a published specification.

Messages that do not have a `Content-Type` field in their header are displayed by user agents as if `Content-Type: Text/plain; Charset=US-ASCII` had been specified.

When a mail reader encounters mail with an unknown `Content-Type` value, it will generally treat it as equivalent to `application/octet-stream`.

The Content-Transfer-Encoding Header Field

Many Content-Types which could usefully be transported via e-mail are represented, in their "natural" format, as 8-bit character or binary data. Such data cannot be transmitted over some transport protocols. For example, SMTP (Simple Mail Transfer Protocol is an Internet standard for transporting e-mail defined by a document called RFC 821) restricts mail messages to 7-bit ASCII data with lines no longer than 1000 characters.

MIME provides two mechanisms for re-encoding such data into a 7-bit short-line format. The `Content-Transfer-Encoding` header field indicates the mechanism used to perform such an encoding. The `Content-Transfer-Encoding` field indicates the transformation that has been used to represent the body in an acceptable manner for transport.

The possible values for the Content-Transfer-Encoding field are:

```
BASE64
QUOTED-PRINTABLE
8BIT
7BIT
BINARY
x-EncodingName
```

These values are not case sensitive. That is, `Base64`, `BASE64` and `bAsE64` are all equivalent. An encoding type of `7BIT` requires that the body is already in a 7-bit mail-ready representation. That is the default value: `Content-Transfer-Encoding: 7BIT` is assumed if the `Content-Transfer-Encoding` header field is not present.

Both `BASE64` and the `QUOTED-PRINTABLE` imply an encoding that consists of lines no longer than 76 ASCII characters. In other respects the two encoding schemes are very different.

The encoding scheme implied by `QUOTED-PRINTABLE` is most appropriate for data that consists primarily of printable ASCII characters. Using this encoding method, printable ASCII character are represented as themselves. The equals sign (=) serves as an escape character. Any character that is not a printable or white space ASCII character is

represented as an equals sign followed by two hexadecimal digits. An equals sign in the message is also represented in this way. Lines that are longer than 76 characters are cut off after the 75th character and the line ends with a equals sign.

The advantages of using the QUOTED-PRINTABLE encoding for message that are mostly printable ASCII characters are that few additional characters are required and the message can be read by human beings who to not have a MIME aware mail reading program. As an example, here is an EDI interchange in QUOTED-PRINTABLE encoding:

```
ISA*00*          *00*          *01*987654321      *12*8005551234     *910=
607*0111*U*00200*110000777*0*T*>
GS*PO*987654321*8005551234*920501*2032*7721*X*002003
ST*850*000000001
BEG*00*NE*MS1112**920501**CONTRACT#
REF*IT*8128827763
N1*ST*MAVERICK SYSTEMS
N3*3312 NEW HAMPSHIRE STREET
N4*SAN JOSE*CA*94811
PO1*1*25*EA***VC*TP8MM*CB*TAPE8MM
PO1*2*30*EA***VC*TP1/4*CB*TAPE1/4INCH
PO1*3*125*EA***VC*DSK31/2*CB*DISK35
CTT*3
SE*11*000000001
GE*1*7721
IEA*1*110000777
```

Except for the ISA segment having been wrapped onto two lines, the QUOTED-PRINTABLE encoding of the interchange is identical to its 7BIT representation.

The BASE64 encoding mechanism is well suited for representing binary files. It represents any sequence of three bytes as four printable ASCII characters. The same interchange as shown above but using the BASE64 encoding would look like:

```
SVNBKjAwKiAgICAgICAgICAqMDAqICAgICAgICAgICowMSo5ODc2NTQzMjEgICAgICAqMTIq
ODAwNTU1MTIzNCAgICAgKjkxMDYwNyowMTExKlUqMDAyMDAqMTEwMDAwNzc3KjAqVCo+CkdT
KlBPKjk4NzY1NDMyMSo4MDA1NTUxMjM0KjkyMDUwMSoyMDMyKjc3MjEqWCowMDIwMDMKU1Qq
ODUwKjAwMDAwMDAwMQpCRUcqMDAqTkUqTVMxMTEyKio5MjA1MDEqKkNPTlRSQUNUIwpSRUYq
SVQqODEyODgyNzc2MwpOMSpTVCpNQVZFUklDSyBTWVNURU1TCk4zKjMzMTIgTkVXIEhBTVBT
SElSRSBTVFJFRVQKTjQqU0FOIEpPU0UqQ0EqOTQ4MTEKUE8xKjEqMjUqRUEqKipWQypUUDhN
TSpDQipUQVBFOE1NClBPMSoyKjMwKkVBKioqVkMqVFAxLzQqQ0IqVEFQRTEvNElOQ0gKUE8x
KjMqMTI1KkVBKioqVkMqRFNLMzEvMipDQipESVNLMzUKQ1RUKjMKU0UqMTEqMDAwMDAwMDAx
CkdFKjEqNzcyMQpJRUEqMSoxMTAwMDA3NzcK
```

BASE64 bears some resemblance to uuencode in both appearance and function. However, uuencode uses characters that may not be processed properly by an EBCDIC gateway.

The values 8bit, 7bit, and binary all imply that **no** encoding has been performed. However, they are useful to indicate of the kind of data contained in the object, and therefore of the kind of encoding that might need to be performed for transmission in a given transport system. 7bit means that the data is all represented as short lines of ASCII data. 8bit means that the lines are short, but there may be non-ASCII characters. Binary means that not only may non-ASCII characters be present, but also that the lines are not necessarily short enough for SMTP transport.

The difference between 8bit and binary is that binary does not require adherence to any limits on line length. 8bit and binary are intended for compatibility with future Internet e-mail transport standards and with gateways to non-Internet environments. As of this writing there are no standardized Internet e-mail transports for which it is legitimate to include unencoded 8-bit or binary data in mail bodies.

Note that the five values defined for the Content-Transfer-Encoding field imply nothing about the Content-Type other than the algorithm by which it was encoded or the transport system requirements if unencoded.

Some implementations may support additional Content-Transfer-Encoding values (it is permitted but strongly discouraged by the standard). Any such additional values must have names that begin with X- to indicate its non-standard status For example: Content-Transfer-Encoding: x-my-new-encoding.

If a Content-Transfer-Encoding header field appears as part of a message header, it applies to the entire body of that message. If a Content-Transfer-Encoding header field appears as part of a body part's headers, it applies only to the body of that body part. If a message or body part is of type Multipart or Message, the Content-Transfer-Encoding must be 7bit, 8bit or Binary.

The encoding mechanisms defined here explicitly encode all data in ASCII. Thus, for example, suppose a message or body part has header fields such as:

```
Content-Type: text/plain; charset=ISO-8859-1
Content-transfer-encoding: base64
```

This should be interpreted to mean that the body is a Base64 ASCII encoding of data that was originally in ISO-8859-1, and will be in that character set again after decoding.

Optional Content-ID Header Field

It may be desirable to allow one body to reference another. Accordingly, bodies may be labeled using the Content-ID header field, which is syntactically identical to the RFC 822 Message-ID header field: Content-ID values should be be as unique as possible.

Optional Content-Description Header Field

The ability to associate descriptive information with a body is often desirable. For example, it may be useful to mark an Image body as “a picture of the Space Shuttle Endeavor.” Such text may be placed in the Content-Description header field.

Summary

Using MIME-Version, Content-Type, and Content-Transfer-Encoding header fields, it is possible to include arbitrary types of data objects in RFC 822 conformant mail messages. No restrictions imposed by RFC 821 or RFC 822 are violated. MIME has been designed to avoid problems caused by additional restrictions imposed by some Internet mail transport mechanisms. The Multipart and Message content types allow mixing and hierarchical structuring of objects of different types in a single message. Further content types provide a mechanism for tagging messages or body parts as audio, image, or other kinds of data. A parameter syntax allows further specification of data format details, particularly the specification of alternate character sets. Additional optional header fields provide mechanisms for certain extensions deemed desirable by many implementors. Finally, a number of useful content types are defined for general use by consenting user agents, notably Text/Richtext, Message/Partial, and Message/External-Body.

To promote interoperability between user agents, the MIME standard specifies a minimal subset of MIME features a user agent must support to be considered MIME conformant.

A Complex Multipart Example

The outline of a complex multipart message follows. This message has five parts to be displayed serially: two introductory plain text parts, an embedded multipart message, a richtext part, and a closing encapsulated text message in a non-ASCII character set. The embedded multipart message has two parts to be displayed in parallel, a picture and an audio fragment.

```
MIME-Version: 1.0
From: Nathaniel Borenstein <nsb@bellcore.com>
Subject: A multipart example
Content-Type: multipart/mixed;
              boundary=unique-boundary-1

This is the preamble area of a multipart message.  Mail readers that
understand multipart format should ignore this preamble.
If you are reading this text, you might want to consider changing to
a mail reader that understands how to properly display multipart
messages.
--unique-boundary-1

Some text appears here...
[Note that the preceding blank line means
no header fields were given and this is text,
with charset US ASCII.  It could have been
done with explicit typing as in the next part.]

--unique-boundary-1
Content-type: text/plain; charset=US-ASCII

This could have been part of the previous part, but illustrates
explicit versus implicit typing of body parts.
```

```
--unique-boundary-1
Content-Type: multipart/parallel; boundary=unique-boundary-2


--unique-boundary-2
Content-Type: audio/basic
Content-Transfer-Encoding: base64

… base64-encoded 8000 Hz single-channel
u-law-format audio data goes here …

--unique-boundary-2
Content-Type: image/gif
Content-Transfer-Encoding: Base64

… base64-encoded image data goes here…

--unique-boundary-2--

--unique-boundary-1
Content-type: text/richtext

This is <bold><italic>richtext.</italic></bold><nl><nl>Isn't it
<bigger><bigger>cool?</bigger></bigger>

--unique-boundary-1
Content-Type: message/rfc822

From: (name in US-ASCII)
Subject: (subject in US-ASCII)
Content-Type: Text/plain; charset=ISO-8859-1
Content-Transfer-Encoding: Quoted-printable

… Additional text in ISO-8859-1 goes here …

--unique-boundary-1--
```