分散システム特論

MIME (Multipurpose Internet Mail Extensions)

[RFC2045] ~ [RFC2049]

情報工学専攻 修士 1 年 2IE01082G 矢野 大介

提出日：2002.2.1

1 概要

インターネットの全ての電子メールは，RFC822 に記述されている単純なテキストメッセージフォーマットに基づいている．このフォーマットでは，メールメッセージはテキストの行から構成されることが決められている．RFC2045 – RFC2049 は，MIME(Multipurpose Internet Mail Extensions) と総称され，単純なテキストメッセージフォーマットを拡張し，以下のことが可能になるように再定義している．

(1) US-ASCII 以外の文字セットによるテキストメッセージボディ
(2) 非テキストのための拡張フォーマット
(3) マルチパートメッセージボディ
(4) US-ASCII 以外の文字セットによるテキストヘッダ情報

1 番目の文書 RFC2045 では，MIME メッセージの構造を記述するために用いる様々なメッセージヘッダを仕様化している．2 番目の RFC2046 では，MIME メディアタイプとその初期値について定義している．3 番目の RFC2047 では，非 US-ASCII テキストのためのインターネットメールヘッダの拡張を定義している．4 番目の RFC2048 では，MIME 関連の機能のための様々な IANA 登録手続を仕様化している．最後に，5 番目の RFC2049 では，MIME 適合条件と MIME メッセージフォーマットの例を記述している．

2 RFC822

2.1 テキストメッセージフォーマット

インターネットテキストメッセージのフォーマットは，たくさんのヘッダフィールドと本文から構成される．また，ヘッダフィールドと本文は空行で区切られる．この場合，空行とは ASCII の CR と LF だけの行をいう．

ヘッダフィールドのフォーマットは，コロンで区切られた名前–値のペアで構成され，本文は任意の文字数の ASCII テキストである (本文はオプションであり，形式上はなくてもかまわない)．RFC822 では本文の行の長さについての制限はないが，RFC821 では，1 行あたり ASCII 文字で 1000 文字 (CRLF を含む) に制限されている．ただし，MTA(Mail Transfer Agent) や MUA(Mail User Agent) が長い行をうまく扱えないので，問題を避けるために 1 行あたりの文字数は 80 字以内に収める慣習がある．

2.2 ヘッダ

主なヘッダは，必須ヘッダ，オプションヘッダ，動的なヘッダ (一般的には MTA だけが作成するヘッダ)，ユーザ定義ヘッダがある．必須ヘッダには，Date，To，From，があり，オプションヘッダには，Cc，Bcc，Subject，Reply-to などがある．また動的なヘッダには，Return-Path，Rceived，Message-ID がある．ユーザ定義ヘッダは，“X-” で始まる ASCII 文字 (コロンを除く 0x21 ~ 0x7E) である．

これらのヘッダにおける注意事項として，ヘッダ値が長い場合，ASCII 文字の単一行として長いまま (256 文字以内) 送るか，複数行に折り返す (1 行あたり 72 文字以内) 必要がある．折り返す場合は，CRLF とそれに続くいくつかの空白文字のシーケンスを行の途中にセパレータとしてはさまなければならない．

3 RFC2045 – RFC2049

3.1 MIME

以前の単純なテキストによるインターネットメールでは，7 ビット ASCII コードしか送信できないテキストベースのメールシステムであったため，

- ワープロなどのアプリケーションデータ (バイナリデータ)
- 日本語などの英語以外の言語コード (8 ビットデータ)

といったデータを含めることができなかった．

そこで，単純なテキストによるインターネットメールシステムを拡張した仕様，MIME が公開された．MIME は，バイナリデータや 8 ビットデータなどのコンテンツを送受信可能にする仕組みである．MIME では，コンテンツをタイプで識別し，アプリケーションや言語と関連付けて格納する仕組みや，コンテンツを ASCII サブセットのテキストにエンコードする仕様を定義している．つまり，MIME によってあらゆるデータをテキスト (7 ビットコード) に変換して送受信するという考えである．

3.2 MIME メッセージフォーマット

基本的に 2.1 節と同様であるが，以下のような異なる点を持つ．

(1) いくつかの特別な MIME に関わるヘッダを持つ

(2) ある約束事に従った本文を持つ

(A) マルチパートメッセージのとき，MIME バウンダリ呼ばれる特別な文字列によっていくつかのパートにわかれている

(B) 各パートの先頭には MIME ヘッダが付加されている

(C) 各パートが ASCII エンコードされたデータを持つ

というように，“パート” を基本とした構造になる．

3.3 MIME ヘッダフィールド

MIME ヘッダには，MIME メッセージヘッダと MIME パートヘッダの 2 種類がある．MIME メッセージヘッダは，RFC822 形式のメッセージヘッダに追加されたヘッダである．具体的には，以下のものがある．

- MIME-Version
- Content-Type
- Content-Transfer-Encoding
- Content-ID
- Content-Description
- Content-Disposition

MIME パートヘッダは，マルチパートメッセージの各本文パートの先頭に置かれ，各パートのコンテンツについての情報を記述する．具体的には，MIME-Version 以外の MIME メッセージヘッダ (上記) である．

MIME メッセージヘッダはメッセージ全体に適用され，MIME パートヘッダは当該パートのみに適用される．以下に，これらの MIME ヘッダフィールドに関して，RFC での定義を示す．

3.3.1 MIME-Version

メッセージの作成に使用された MIME のバージョンを表す．現時点では，使用されている MIME バージョンは 1.0 以外には存在しない．

3.3.2 Content-Type

Content-Type の目的は MUA がメッセージを解析し，メッセージに含まれるデータのタイプを識別し，正しく表示できるようにすることである．このフィールド値はメディアタイプと呼ばれ，RFC2046 で定義されている．メディアタイプには，メディアタイプとサブタイプがある．Content-Type フィールドのフォーマットは次のようになる．

`Content-Type: メディアタイプ/サブタイプ [; パラメータ名=パラメータ値]`

- メディアタイプ
  メディアタイプには以下のようなものがある．
  (1) Discrete Media Type
    (A) text
    (B) image
    (C) audio
    (D) video
    (E) application
  (2) Composite Media Type
    (A) multipart
    (B) message
- サブタイプ
  サブタイプはあるメディアタイプに特徴的な書式を特定する．また各メディアタイプには 1 つまたはそれ以上のサブタイプが定義されている．新しいサブタイプを定義するには，以下の方法がある．
  (1) RFC2048 に従って，IANA(Internet Assigned Number Authority) に登録する
  (2) 外部登録を行わずに，“X-” で始まる値を使用する．
- パラメータ
  パラメータはメッセージへの付加的な情報を表す．大抵はある特定のサブタイプに関連付けられているが，メディアタイプによりパラメータを定義する場合もある．例えば，multipart メディアタイプを使用する場合には，必ず boundary パラメータを指定する．

ここで，Composite Media Type(multipart ，message) で指定される MIME バウンダリについて説明しておく．MIME バウンダリは，MIME メッセージ内でのメッセージパート間の境界を定義するものであり，7 ビット US-ASCII テキストで記述された文字列である．また，メッセージ内で一意な文字列でなければならない．例えば以下のように指定する．

```
Content-Type: multipart/mixed; boundary="boundary_str"
```

1 つのパートの開始を意味する境界識別子は，2 つのハイフンを始めにつけて，

```
--boundary_str
```

全体のマルチーパートの終了を意味する識別子は，2 つのハイフンを前後につけて，

```
--boundary_str--
```

となる．また，バウンダリに挟まれた各パートは，さらにそのヘッダ中でバウンダリを定義してパートをネストすることができる．

3.3.3 Content-Transfer-Encoding

このフィールドはメッセージまたはメッセージパートのエンコード方式を表す．Content-Transfer-Encoding のフォーマットは以下のようになる．

Content-Transfer-Encoding: エンコード方式

エンコード方式には，以下のものがある．

(1) 7bit
(2) 8bit
(3) binary
(4) quoted-printable
(5) base64

注意事項として，Content-Type が multipart，message の場合は，7bit，8bit，binary 以外の方式をとることができない．これは，エンコードのネストが生じて効率が落ちることを防ぐためである．格納するデータをエンコードする必要がある場合は，multipart や message 全体ではなく，最内部パートでエンコードしなければならない．

3.3.4 Content-ID

Content-ID ヘッダによってあるメッセージの本文から別のメッセージの本文を参照できる．このフィールド値は Message-ID と同じく，世界中で一意でなければならない．Content-ID は必須ではないが，messege/external-body(メッセージパートの実体がメッセージ本文内ではなくネットワーク上の別の場所にある) の場合には，相互に参照するために Content-ID ヘッダの使用が必須である．

3.3.5 Content-Description

Content-Description ヘッダによって，ASCII テキストによるメッセージのコンテンツの説明をつけることができる．Content-Description ヘッダは常に選択的である．また，説明の文字コードのデフォルトは US-ASCII であるが，特別な文字エンコードメカニズムにより，他の文字セットでも記述できる．ヘッダでの非 ASCII 文字の利用については RFC2047 に記述がある．その場合のエンコード形式を以下に示す．

=?文字コード?エンコード方式?エンコード済みテキスト?=

エンコード部分は，“=?” と “?=” で囲まれ，“?” によって 3 つのブロックに分かれている．先頭ブロックは，エンコードされたデータの文字コードを示している．次のブロックは，元データのエンコード形式を示している．ここが，“B” ならば base64，“Q” ならば quoted-printable であることを示す．最後のブロックは，エンコード方式に従ったエンコード済のデータである．

例えば，“日本語” をこの形式でエンコードすると，

=?iso-2022-jp?B?GyRcRnxLXDhsGyhC?=

となる．

3.3.6 Content-Disposition(RFC2183)

Content-Disposition ヘッダは，本文に含まれるファイルの提示方法についてのヒントを与える．フォーマットは以下のようになる

Content-Disposition: タイプ [; パラメータ名=パラメータ値]

タイプに以下のものがある

(1) inline
(2) attachment

パラメータには，filename などがあり，filename はオリジナルの添付ファイル名を伝えることができる．filename の他には，create-date や modification-date などがある．

3.4 MIME エンコード

インターネットメール (SMTP) の制限に従うために，任意のバイナリデータまたは非 US-ASCII 文字セットのテキストデータは，ASCII テキストにエンコードする必要がある．ここでは，3.3.3 節で示したエンコード方式を以下のように分類し，説明する．

(1) 7bit，8bit，binary
(2) quoted-printable
(3) base64

3.4.1 7bit，8bit，binary

この方式では，コンテンツは全くエンコードが行われておらず，値が示す通りのタイプのデータが含まれていることを示している．しかし，インターネットメールシステムを介した配送において安全であることが保証されているのは 7 ビットデータのみであり，8bit や binary は使われるべきではない．これは，メッセージが，RFC821 準拠のメールシステムの途中の MTA やゲートウェイなどを通過して配信される間に，データの 8 ビット目が強制的に削除され (0 にされ) ることがあるためである．この結果，“ビット落ち” という現象が発生し，文字化けの原因となる．

では，なぜエンコード方式に “8bit” や “binary” が存在するのかというと，インターネットメールを 8 ビットデータに対応させるという動きがあるためで，その際に 8 ビットのままで利用できるようにするための定義である．しかし，このような活動は限定的なものだるため，やはり使うべきではない．

3.4.2 quoted-printable

quoted-printable は，人間が判読できる (ASCII テキスト) 形式だが 7 ビットデータでないデータを 7 ビットデータに変換するエンコード方式である．例えば，ヨーロッパやロシアなどの言語を表現するとき，ほとんどが ASCII で表現できるため，便利である．

3.4.3 base64

base64 は，7bit または quoted-printable によるエンコードが向かないファイル (主にバイナリデータ) をインターネットメールシステムを経由して送る必要がある場合に広く使用されている．結果的に base64 でもデータは 7 ビットデータに変換される．しかし，7 ビットデータのまま送信する場合や，7 ビットでないテキストデータを quoted-printable でエンコードして送信する場合に比べて効率が悪くなる．また，エンコードされたデータのサイズは，オリジナルデータより 33%増大する．

4 実装

MIME を実装しているフリーソフトウェアには，X Window System 上で動作する電子メールクライアント Sylpeed (http://sylpheed.good-day.net/) がある．

発表では添付パート作成部分のソースを示した．Sylpheed では，Content-Transfer-Encoding が，7bit でなければ，base64 でエンコードする実装になっている．つまり，デフォルトでは，それ以外の 8bit，binary，quoted-printable は使われない．

5 補足

5.1 uuencode と base64 の違い

uuencode でデータの符号化に使用される文字は，すべて US-ASCII となっている．uuencode では取り出した数値に 0x20(空白) を加算して文字にしており，符号化されたテキストからデータを取り出すには，文字から 0x20 を引けばよい．つまり，標準的な uuencode は符号化された文字に空白が存在する．そして，メールゲートウェイによっては，メール内に存在する余分な空白文字を削除することがある．したがって，uuencode で符号化されたデータをメールで送信すると，どこかでデータが欠落してしまう可能性がある．

base64 は uuencode と同様に，3 バイトを一つの単位として 4 文字を生成する．しかし，uuencode とは異なり，符号化された文字には ”行” ごとの保存は行わない．また，base64 では改行文字や空白文字は無視される．base64 では，数値に対応する英数文字に変換するため，ASCII とは違う文字コードを使用しているコンピュータでもバイナリデータを取り出すことがでる．

これらのことから uuencode は，インターネットメールにおけるエンコード方式として適していないことが，RFC2045 で述べられている．