Network Working Group
Request for Comments: 2047
Obsoletes: 1521, 1522, 1590
Category: Standards Track

K. Moore
University of Tennessee
November 1996

MIME (Multipurpose Internet Mail Extensions) Part Three:

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Abstract

STD 11, RFC 822, defines a message representation protocol specifying considerable detail about US-ASCII message headers, and leaves the message content, or message body, as flat US-ASCII text. This set of documents, collectively called the Multipurpose Internet Mail Extensions, or MIME, redefines the format of messages to allow for

1. textual message bodies in character sets other than US-ASCII,
2. an extensible set of different formats for non-textual message bodies,
3. multi-part message bodies, and
4. textual header information in character sets other than US-ASCII.

These documents are based on earlier work documented in RFC 934, STD 11, and RFC 1049, but extends and revises them. Because RFC 822 said so little about message bodies, these documents are largely orthogonal to (rather than a revision of) RFC 822.

This particular document is the third document in the series. It describes extensions to RFC 822 to allow non-US-ASCII text data in Internet mail header fields.

Other documents in this series include:

- RFC 2045, which specifies the various headers used to describe the structure of MIME messages.
- RFC 2046, which defines the general structure of the MIME media typing system and defines an initial set of media types,
- RFC 2048, which specifies various IANA registration procedures for MIME-related facilities, and
- RFC 2049, which describes MIME conformance criteria and provides some illustrative examples of MIME message formats, acknowledgements, and the bibliography.

These documents are revisions of RFCs 1521, 1522, and 1590, which themselves were revisions of RFCs 1341 and 1342. An appendix in RFC 2049 describes differences and changes from previous versions.

1. Introduction

RFC 2045 describes a mechanism for denoting textual body parts which are coded in various character sets, as well as methods for encoding such body parts as sequences of printable US-ASCII characters. This memo describes similar techniques to allow the encoding of non-ASCII text in various portions of a RFC 822 [2] message header, in a manner which is unlikely to confuse existing message handling software.

Like the encoding techniques described in RFC 2045, the techniques outlined here were designed to allow the use of non-ASCII characters in message headers in a way which is unlikely to be disturbed by the quirks of existing Internet mail handling programs. In particular, some mail relaying programs are known to (a) delete some message header fields while retaining others, (b) rearrange the order of addresses in To or Cc fields, (c) rearrange the (vertical) order of header fields, and/or (d) "wrap" message headers at different places than those in the original message. In addition, some mail reading programs are known to have difficulty correctly parsing message headers which, while legal according to RFC 822, make use of backslash-quoting to "hide" special characters such as "<", ",", or ":", or which exploit other infrequently-used features of that specification.

While it is unfortunate that these programs do not correctly interpret RFC 822 headers, to "break" these programs would cause severe operational problems for the Internet mail system. The extensions described in this memo therefore do not rely on little-used features of RFC 822.

Instead, certain sequences of "ordinary" printable ASCII characters (known as "encoded-words") are reserved for use as encoded data. The syntax of encoded-words is such that they are unlikely to "accidentally" appear as normal text in message headers. Furthermore, the characters used in encoded-words are restricted to those which do not have special meanings in the context in which the encoded-word appears.

Generally, an "encoded-word" is a sequence of printable ASCII characters that begins with "=?", ends with "?=", and has two "?"s in between. It specifies a character set and an encoding method, and also includes the original text encoded as graphic ASCII characters, according to the rules for that encoding method.

A mail composer that implements this specification will provide a means of inputting non-ASCII text in header fields, but will translate these fields (or appropriate portions of these fields) into encoded-words before inserting them into the message header.

A mail reader that implements this specification will recognize encoded-words when they appear in certain portions of the message header. Instead of displaying the encoded-word "as is", it will reverse the encoding and display the original text in the designated character set.

NOTES

This memo relies heavily on notation and terms defined RFC 822 and RFC 2045. In particular, the syntax for the ABNF used in this memo is defined in RFC 822, as well as many of the terminal or nonterminal symbols from RFC 822 are used in the grammar for the header extensions defined here. Among the symbols defined in RFC 822 and referenced in this memo are: 'addr-spec', 'atom', 'CHAR', 'comment', 'CTLs', 'ctext', 'linear-white-space', 'phrase', 'quoted-pair'. 'quoted-string', 'SPACE', and 'word'. Successful implementation of this protocol extension requires careful attention to the RFC 822 definitions of these terms.

When the term "ASCII" appears in this memo, it refers to the "7-Bit American Standard Code for Information Interchange", ANSI X3.4-1986. The MIME charset name for this character set is "US-ASCII". When not specifically referring to the MIME charset name, this document uses the term "ASCII", both for brevity and for consistency with RFC 822. However, implementors are warned that the character set name must be spelled "US-ASCII" in MIME message and body part headers.

This memo specifies a protocol for the representation of non-ASCII text in message headers. It specifically DOES NOT define any translation between "8-bit headers" and pure ASCII headers, nor is any such translation assumed to be possible.

2. Syntax of encoded-words

An 'encoded-word' is defined by the following ABNF grammar. The notation of RFC 822 is used, with the exception that white space characters MUST NOT appear between components of an 'encoded-word'.

```
encoded-word = "=?" charset "?" encoding "?" encoded-text "?="

charset = token    ; see section 3

encoding = token   ; see section 4

token = 1*<Any CHAR except SPACE, CTLs, and especials>

especials = "(" / ")" / "<" / ">" / "@" / "," / ";" / ":" / "
            <"> / "/" / "[" / "]" / "?" / "." / "="

encoded-text = 1*<Any printable ASCII character other than "?"
                  or SPACE>
               ; (but see "Use of encoded-words in message
               ; headers", section 5)
```

Both 'encoding' and 'charset' names are case-independent. Thus the charset name "ISO-8859-1" is equivalent to "iso-8859-1", and the encoding named "Q" may be spelled either "Q" or "q".

An 'encoded-word' may not be more than 75 characters long, including 'charset', 'encoding', 'encoded-text', and delimiters. If it is desirable to encode more text than will fit in an 'encoded-word' of 75 characters, multiple 'encoded-word's (separated by CRLF SPACE) may be used.

While there is no limit to the length of a multiple-line header field, each line of a header field that contains one or more 'encoded-word's is limited to 76 characters.

The length restrictions are included both to ease interoperability through internetwork mail gateways, and to impose a limit on the amount of lookahead a header parser must employ (while looking for a final ?= delimiter) before it can decide whether a token is an "encoded-word" or something else.

IMPORTANT: 'encoded-word's are designed to be recognized as 'atom's by an RFC 822 parser. As a consequence, unencoded white space characters (such as SPACE and HTAB) are FORBIDDEN within an 'encoded-word'. For example, the character sequence

```
=?iso-8859-1?q?this is some text?=
```

would be parsed as four 'atom's, rather than as a single 'atom' (by an RFC 822 parser) or 'encoded-word' (by a parser which understands 'encoded-words'). The correct way to encode the string "this is some text" is to encode the SPACE characters as well, e.g.

```
=?iso-8859-1?q?this=20is=20some=20text?=
```

The characters which may appear in 'encoded-text' are further restricted by the rules in section 5.

3. Character sets

The 'charset' portion of an 'encoded-word' specifies the character set associated with the unencoded text. A 'charset' can be any of the character set names allowed in an MIME "charset" parameter of a "text/plain" body part, or any character set name registered with IANA for use with the MIME text/plain content-type.

Some character sets use code-switching techniques to switch between "ASCII mode" and other modes. If unencoded text in an 'encoded-word' contains a sequence which causes the charset interpreter to switch out of ASCII mode, it MUST contain additional control codes such that ASCII mode is again selected at the end of the 'encoded-word'. (This rule applies separately to each 'encoded-word', including adjacent 'encoded-word's within a single header field.)

When there is a possibility of using more than one character set to represent the text in an 'encoded-word', and in the absence of private agreements between sender and recipients of a message, it is recommended that members of the ISO-8859-* series be used in preference to other character sets.

4. Encodings

Initially, the legal values for "encoding" are "Q" and "B". These encodings are described below. The "Q" encoding is recommended for use when most of the characters to be encoded are in the ASCII character set; otherwise, the "B" encoding should be used. Nevertheless, a mail reader which claims to recognize 'encoded-word's MUST be able to accept either encoding for any character set which it supports.

Only a subset of the printable ASCII characters may be used in 'encoded-text'. Space and tab characters are not allowed, so that the beginning and end of an 'encoded-word' are obvious. The "?" character is used within an 'encoded-word' to separate the various portions of the 'encoded-word' from one another, and thus cannot appear in the 'encoded-text' portion. Other characters are also illegal in certain contexts. For example, an 'encoded-word' in a 'phrase' preceding an address in a From header field may not contain any of the "specials" defined in RFC 822. Finally, certain other characters are disallowed in some contexts, to ensure reliability for messages that pass through internetwork mail gateways.

The "B" encoding automatically meets these requirements. The "Q" encoding allows a wide range of printable characters to be used in non-critical locations in the message header (e.g., Subject), with fewer characters available for use in other locations.

4.1. The "B" encoding

The "B" encoding is identical to the "BASE64" encoding defined by RFC 2045.

4.2. The "Q" encoding

The "Q" encoding is similar to the "Quoted-Printable" content-transfer-encoding defined in RFC 2045. It is designed to allow text containing mostly ASCII characters to be decipherable on an ASCII terminal without decoding.

1. Any 8-bit value may be represented by a "=" followed by two hexadecimal digits. For example, if the character set in use were ISO-8859-1, the "=" character would thus be encoded as "=3D", and a SPACE by "=20". (Upper case should be used for hexadecimal digits "A" through "F".)
2. The 8-bit hexadecimal value 20 (e.g., ISO-8859-1 SPACE) may be represented as "_" (underscore, ASCII 95.). (This character may not pass through some internetwork mail gateways, but its use will greatly enhance readability of "Q" encoded data with mail readers that do not support this encoding.) Note that the "_" always represents hexadecimal 20, even if the SPACE character occupies a different code position in the character set in use.
3. 8-bit values which correspond to printable ASCII characters other than "=", "?", and "_" (underscore), MAY be represented as those characters. (But see section 5 for restrictions.) In particular, SPACE and TAB MUST NOT be represented as themselves within encoded words.

5. Use of encoded-words in message headers

An 'encoded-word' may appear in a message header or body part header according to the following rules:

1. An 'encoded-word' may replace a 'text' token (as defined by RFC 822) in any Subject or Comments header field, any extension message header field, or any MIME body part field for which the field body is defined as '*text'. An 'encoded-word' may also appear in any user-defined ("X-") message or body part header field.

   Ordinary ASCII text and 'encoded-word's may appear together in the same header field. However, an 'encoded-word' that appears in a header field defined as '*text' MUST be separated from any adjacent 'encoded-word' or 'text' by 'linear-white-space'.

2. An 'encoded-word' may appear within a 'comment' delimited by "(" and ")", i.e., wherever a 'ctext' is allowed. More precisely, the RFC 822 ABNF definition for 'comment' is amended as follows:

   ```
   comment = "(" *(ctext / quoted-pair / comment / encoded-word) ")"
   ```

   A "Q"-encoded 'encoded-word' which appears in a 'comment' MUST NOT contain the characters "(", ")" or " 'encoded-word' that appears in a 'comment' MUST be separated from any adjacent 'encoded-word' or 'ctext' by 'linear-white-space'.

   It is important to note that 'comment's are only recognized inside "structured" field bodies. In fields whose bodies are defined as '*text', "(" and ")" are treated as ordinary characters rather than comment delimiters, and rule (1) of this section applies. (See RFC 822, sections 3.1.2 and 3.1.3)

3. As a replacement for a 'word' entity within a 'phrase', for example, one that precedes an address in a From, To, or Cc header. The ABNF definition for 'phrase' from RFC 822 thus becomes:

   ```
   phrase = 1*( encoded-word / word )
   ```

   In this case the set of characters that may be used in a "Q"-encoded 'encoded-word' is restricted to: <upper and lower case ASCII letters, decimal digits, "!", "*", "+", "-", "/", "=", and "_" (underscore, ASCII 95.)>. An 'encoded-word' that appears within a 'phrase' MUST be separated from any adjacent 'word', 'text' or 'special' by 'linear-white-space'.

These are the ONLY locations where an 'encoded-word' may appear. In particular:

- An 'encoded-word' MUST NOT appear in any portion of an 'addr-spec'.
- An 'encoded-word' MUST NOT appear within a 'quoted-string'.
- An 'encoded-word' MUST NOT be used in a Received header field.
- An 'encoded-word' MUST NOT be used in parameter of a MIME Content-Type or Content-Disposition field, or in any structured field body except within a 'comment' or 'phrase'.

The 'encoded-text' in an 'encoded-word' must be self-contained; 'encoded-text' MUST NOT be continued from one 'encoded-word' to another. This implies that the 'encoded-text' portion of a "B" 'encoded-word' will be a multiple of 4 characters long; for a "Q" 'encoded-word', any "=" character that appears in the 'encoded-text' portion will be followed by two hexadecimal characters.

Each 'encoded-word' MUST encode an integral number of octets. The 'encoded-text' in each 'encoded-word' must be well-formed according to the encoding specified; the 'encoded-text' may not be continued in the next 'encoded-word'. (For example, "=?charset?Q?=?= =?charset?Q?AB?=" would be illegal, because the two hex digits "AB" must follow the "=" in the same 'encoded-word'.)

Each 'encoded-word' MUST represent an integral number of characters. A multi-octet character may not be split across adjacent 'encoded-word's.

Only printable and white space character data should be encoded using this scheme. However, since these encoding schemes allow the encoding of arbitrary octet values, mail readers that implement this decoding should also ensure that display of the decoded data on the recipient's terminal will not cause unwanted side-effects.

Use of these methods to encode non-textual data (e.g., pictures or sounds) is not defined by this memo. Use of 'encoded-word's to represent strings of purely ASCII characters is allowed, but discouraged. In rare cases it may be necessary to encode ordinary text that looks like an 'encoded-word'.

6. Support of 'encoded-word's by mail readers

6.1. Recognition of 'encoded-word's in message headers

A mail reader must parse the message and body part headers according to the rules in RFC 822 to correctly recognize 'encoded-word's.

'encoded-word's are to be recognized as follows:

1. Any message or body part header field defined as '*text', or any user-defined header field, should be parsed as follows: Beginning at the start of the field-body and immediately following each occurrence of 'linear-white-space', each sequence of up to 75 printable characters (not containing any 'linear-white-space') should be examined to see if it is an 'encoded-word' according to the syntax rules in section 2. Any other sequence of printable characters should be treated as ordinary ASCII text.
2. Any header field not defined as '*text' should be parsed according to the syntax rules for that header field. However, any 'word' that appears within a 'phrase' should be treated as an 'encoded-word' if it meets the syntax rules in section 2. Otherwise it should be treated as an ordinary 'word'.
3. Within a 'comment', any sequence of up to 75 printable characters (not containing 'linear-white-space'), that meets the syntax rules in section 2, should be treated as an 'encoded-word'. Otherwise it should be treated as normal comment text.
4. A MIME-Version header field is NOT required to be present for 'encoded-word's to be interpreted according to this specification. One reason for this is that the mail reader is not expected to parse the entire message header before displaying lines that may contain 'encoded-word's.

6.2. Display of 'encoded-word's

Any 'encoded-word's so recognized are decoded, and if possible, the resulting unencoded text is displayed in the original character set.

NOTE: Decoding and display of encoded-words occurs *after* a structured field body is parsed into tokens. It is therefore possible to hide 'special' characters in encoded-words which, when displayed, will be indistinguishable from 'special' characters in the surrounding text. For this and other reasons, it is NOT generally possible to translate a message header containing 'encoded-word's to an unencoded form which can be parsed by an RFC 822 mail reader.

When displaying a particular header field that contains multiple 'encoded-word's, any 'linear-white-space' that separates a pair of adjacent 'encoded-word's is ignored. (This is to allow the use of multiple 'encoded-word's to represent long strings of unencoded text, without having to separate 'encoded-word's where spaces occur in the unencoded text.)

In the event other encodings are defined in the future, and the mail reader does not support the encoding used, it may either (a) display the 'encoded-word' as ordinary text, or (b) substitute an appropriate message indicating that the text could not be decoded.

If the mail reader does not support the character set used, it may (a) display the 'encoded-word' as ordinary text (i.e., as it appears in the header), (b) make a "best effort" to display using such characters as are available, or (c) substitute an appropriate message indicating that the decoded text could not be displayed.

If the character set being used employs code-switching techniques, display of the encoded text implicitly begins in "ASCII mode". In addition, the mail reader must ensure that the output device is once again in "ASCII mode" after the 'encoded-word' is displayed.

6.3. Mail reader handling of incorrectly formed 'encoded-word's

It is possible that an 'encoded-word' that is legal according to the syntax defined in section 2, is incorrectly formed according to the rules for the encoding being used. For example:

1. An 'encoded-word' which contains characters which are not legal for a particular encoding (for example, a "-" in the "B" encoding, or a SPACE or HTAB in either the "B" or "Q" encoding), is incorrectly formed.
2. Any 'encoded-word' which encodes a non-integral number of characters or octets is incorrectly formed.

A mail reader need not attempt to display the text associated with an 'encoded-word' that is incorrectly formed. However, a mail reader MUST NOT prevent the display or handling of a message because an 'encoded-word' is incorrectly formed.

7. Conformance

A mail composing program claiming compliance with this specification MUST ensure that any string of non-white-space printable ASCII characters within a '*text' or '*ctext' that begins with "=?" and ends with "?=" be a valid 'encoded-word'. ("begins" means: at the start of the field-body, immediately following 'linear-white-space', or immediately following a "(" for an 'encoded-word' within '*ctext'; "ends" means: at the end of the field-body, immediately preceding 'linear-white-space', or immediately preceding a ")" for an 'encoded-word' within '*ctext'.) In addition, any 'word' within a 'phrase' that begins with "=?" and ends with "?=" must be a valid 'encoded-word'.

A mail reading program claiming compliance with this specification must be able to distinguish 'encoded-word's from 'text', 'ctext', or 'word's, according to the rules in section 6, anytime they appear in appropriate places in message headers. It must support both the "B" and "Q" encodings for any character set which it supports. The program must be able to display the unencoded text if the character set is "US-ASCII". For the ISO-8859-* character sets, the mail reading program must at least be able to display the characters which are also in the ASCII set.

8. Examples

The following are examples of message headers containing 'encoded-word's:

```
From: =?US-ASCII?Q?Keith_Moore?= <moore@cs.utk.edu>
To: =?ISO-8859-1?Q?Keld_J=F8rn_Simonsen?= <keld@dkuug.dk>
CC: =?ISO-8859-1?Q?Andr=E9?= Pirard <PIRARD@vm1.ulg.ac.be>
Subject: =?ISO-8859-1?B?SWYgeW91IGNhbiByZWFkIHRoaXMgeW8=?=
 =?ISO-8859-2?B?dSB1bmRlcnN0YW5kIHRoZSBleGFtcGxlLg==?=
```

Note: In the first 'encoded-word' of the Subject field above, the last "=" at the end of the 'encoded-text' is necessary because each 'encoded-word' must be self-contained (the "=" character completes a group of 4 base64 characters representing 2 octets). An additional octet could have been encoded in the first 'encoded-word' (so that the encoded-word would contain an exact multiple of 3 encoded octets), except that the second 'encoded-word' uses a different 'charset' than the first one.

```
From: =?ISO-8859-1?Q?Olle_J=E4rnefors?= <ojarnef@admin.kth.se>
To: ietf-822@dimacs.rutgers.edu, ojarnef@admin.kth.se
Subject: Time for ISO 10646?

To: Dave Crocker <dcrocker@mordor.stanford.edu>
Cc: ietf-822@dimacs.rutgers.edu, paf@comsol.se
From: =?ISO-8859-1?Q?Patrik_F=E4ltstr=F6m?= <paf@nada.kth.se>
Subject: Re: RFC-HDR care and feeding

From: Nathaniel Borenstein <nsb@thumper.bellcore.com>
      (=?iso-8859-8?b?7eXs+SDv4SDp7Oj08A==?=)
To: Greg Vaudreuil <gvaudre@NRI.Reston.VA.US>, Ned Freed
   <ned@innosoft.com>, Keith Moore <moore@cs.utk.edu>
Subject: Test of new header generator
MIME-Version: 1.0
Content-type: text/plain; charset=ISO-8859-1
```

The following examples illustrate how text containing 'encoded-word's which appear in a structured field body. The rules are slightly different for fields defined as '*text' because "(" and ")" are not recognized as 'comment' delimiters. [Section 5, paragraph (1)].

In each of the following examples, if the same sequence were to occur in a '*text' field, the "displayed as" form would NOT be treated as encoded words, but be identical to the "encoded form". This is because each of the encoded-words in the following examples is adjacent to a "(" or ")" character.

```
encoded form                                    displayed as
-------------------------------------------------------------------
(=?ISO-8859-1?Q?a?=)                            (a)

(=?ISO-8859-1?Q?a?= b)                          (a b)
```

Within a 'comment', white space MUST appear between an 'encoded-word' and surrounding text. [Section 5, paragraph (2)]. However, white space is not needed between the initial "(" that begins the 'comment', and the 'encoded-word'.

```
(=?ISO-8859-1?Q?a?= =?ISO-8859-1?Q?b?=)     (ab)
```

White space between adjacent 'encoded-word's is not displayed.

```
(=?ISO-8859-1?Q?a?=  =?ISO-8859-1?Q?b?=)    (ab)
```

Even multiple SPACEs between 'encoded-word's are ignored for the purpose of display.

```
(=?ISO-8859-1?Q?a?=                         (ab)
    =?ISO-8859-1?Q?b?=)
```

Any amount of linear-space-white between 'encoded-word's, even if it includes a CRLF followed by one or more SPACEs, is ignored for the purposes of display.

```
(=?ISO-8859-1?Q?a_b?=)                      (a b)
```

In order to cause a SPACE to be displayed within a portion of encoded text, the SPACE MUST be encoded as part of the 'encoded-word'.

```
(=?ISO-8859-1?Q?a?= =?ISO-8859-2?Q?_b?=)    (a b)
```

In order to cause a SPACE to be displayed between two strings of encoded text, the SPACE MAY be encoded as part of one of the 'encoded-word's.

9. References

[RFC822] Crocker, D., "Standard for the format of ARPA Internet text messages", STD 11, RFC 822, August 1982.

[RFC2049] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Five: Conformance Criteria and Examples", RFC 2049, November 1996.

[RFC2045] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996.

[RFC2046] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, November 1996.

[RFC2048] Freed, N., Klensin, J., and J. Postel, "Multipurpose Internet Mail Extensions (MIME) Part Four: Registration Procedures", RFC 2048, November 1996.

10. Security Considerations

Security issues are not discussed in this memo.

11. Acknowledgements

The author wishes to thank Nathaniel Borenstein, Issac Chan, Lutz Donnerhacke, Paul Eggert, Ned Freed, Andreas M. Kirchwitz, Olle Jarnefors, Mike Rosin, Yutaka Sato, Bart Schaefer, and Kazuhiko Yamamoto, for their helpful advice, insightful comments, and illuminating questions in response to earlier versions of this specification.

12. Author's Address

Keith Moore
University of Tennessee
Computer Science Dept.
107 Ayres Hall
Knoxville, TN 37996-1301
USA
EMail: moore@cs.utk.edu

Appendix - changes since RFC 1522 (in no particular order)

- explicitly state that the MIME-Version is not requried to use 'encoded-word's.
- add explicit note that SPACEs and TABs are not allowed within 'encoded-word's, explaining that an 'encoded-word' must look like an 'atom' to an RFC822 parser.values, to be precise).
- add examples from Olle Jarnefors (thanks!) which illustrate how encoded-words with adjacent linear-white-space are displayed.
- explicitly list terms defined in RFC822 and referenced in this memo
- fix transcription typos that caused one or two lines and a couple of characters to disappear in the resulting text, due to nroff quirks.
- clarify that encoded-words are allowed in '*text' fields in both RFC822 headers and MIME body part headers, but NOT as parameter values.
- clarify the requirement to switch back to ASCII within the encoded portion of an 'encoded-word', for any charset that uses code switching sequences.
- add a note about 'encoded-word's being delimited by "(" and ")" within a comment, but not in a *text (how bizarre!).
- fix the Andre Pirard example to get rid of the trailing "_" after the =E9. (no longer needed post-1342).
- clarification: an 'encoded-word' may appear immediately following the initial "(" or immediately before the final ")" that delimits a comment, not just adjacent to "(" and ")" *within* *ctext.
- add a note to explain that a "B" 'encoded-word' will always have a multiple of 4 characters in the 'encoded-text' portion.
- add note about the "=" in the examples
- note that processing of 'encoded-word's occurs *after* parsing, and some of the implications thereof.
- explicitly state that you can't expect to translate between 1522 and either vanilla 822 or so-called "8-bit headers".
- explicitly state that 'encoded-word's are not valid within a 'quoted-string'.