RFC # 822

Obsoletes: RFC #733 (NIC #41952)

STANDARD FOR THE FORMAT OF

ARPA INTERNET TEXT MESSAGES

August 13, 1982

Revised by

David H. Crocker

Dept. of Electrical Engineering
University of Delaware, Newark, DE 19711
Network: DCrocker @ UDel-Relay

TABLE OF CONTENTS

APPENDIX

PREFACE

By 1977, the Arpanet employed several informal standards for the text messages (mail) sent among its host computers. It was felt necessary to codify these practices and provide for those features that seemed imminent. The result of that effort was Request for Comments (RFC) #733, "Standard for the Format of ARPA Network Text Message", by Crocker, Vittal, Pogran, and Henderson. The specification attempted to avoid major changes in existing software, while permitting several new features.

This document revises the specifications in RFC #733, in order to serve the needs of the larger and more complex ARPA Internet. Some of RFC #733's features failed to gain adequate acceptance. In order to simplify the standard and the software that follows it, these features have been removed. A different addressing scheme is used, to handle the case of inter-network mail; and the concept of re-transmission has been introduced.

This specification is intended for use in the ARPA Internet. However, an attempt has been made to free it of any dependence on that environment, so that it can be applied to other network text message systems.

The specification of RFC #733 took place over the course of one year, using the ARPANET mail environment, itself, to provide an on-going forum for discussing the capabilities to be included. More than twenty individuals, from across the country, participated in the original discussion. The development of this revised specification has, similarly, utilized network mail-based group discussion. Both specification efforts greatly benefited from the comments and ideas of the participants.

The syntax of the standard, in RFC #733, was originally specified in the Backus-Naur Form (BNF) meta-language. Ken L. Harrenstien, of SRI International, was responsible for re-coding the BNF into an augmented BNF that makes the representation smaller and easier to understand.

1. INTRODUCTION

1.1. SCOPE

This standard specifies a syntax for text messages that are sent among computer users, within the framework of "electronic mail". The standard supersedes the one specified in ARPANET Request for Comments #733, "Standard for the Format of ARPA Network Text Messages".

In this context, messages are viewed as having an envelope and contents. The envelope contains whatever information is needed to accomplish transmission and delivery. The contents compose the object to be delivered to the recipient. This standard applies only to the format and some of the semantics of message contents. It contains no specification of the information in the envelope.

However, some message systems may use information from the contents to create the envelope. It is intended that this standard facilitate the acquisition of such information by programs.

Some message systems may store messages in formats that differ from the one specified in this standard. This specification is intended strictly as a definition of what message content format is to be passed BETWEEN hosts.

Note: This standard is NOT intended to dictate the internal formats used by sites, the specific message system features that they are expected to support, or any of the characteristics of user interface programs that create or read messages.

A distinction should be made between what the specification REQUIRES and what it ALLOWS. Messages can be made complex and rich with formally-structured components of information or can be kept small and simple, with a minimum of such information. Also, the standard simplifies the interpretation of differing visual formats in messages; only the visual aspect of a message is affected and not the interpretation of information within it. Implementors may choose to retain such visual distinctions.

The formal definition is divided into four levels. The bottom level describes the meta-notation used in this document. The second level describes basic lexical analyzers that feed tokens to higher-level parsers. Next is an overall specification for messages; it permits distinguishing individual fields. Finally, there is definition of the contents of several structured fields.

1.2. COMMUNICATION FRAMEWORK

Messages consist of lines of text. No special provisions are made for encoding drawings, facsimile, speech, or structured text. No significant consideration has been given to questions of data compression or to transmission and storage efficiency, and the standard tends to be free with the number of bits consumed. For example, field names are specified as free text, rather than special terse codes.

A general "memo" framework is used. That is, a message consists of some information in a rigid format, followed by the main part of the message, with a format that is not specified in this document. The syntax of several fields of the rigidly-formated ("headers") section is defined in this specification; some of these fields must be included in all messages.

The syntax that distinguishes between header fields is specified separately from the internal syntax for particular fields. This separation is intended to allow simple parsers to operate on the general structure of messages, without concern for the detailed structure of individual header fields. Appendix B is provided to facilitate construction of these parsers.

In addition to the fields specified in this document, it is expected that other fields will gain common use. As necessary, the specifications for these "extension-fields" will be published through the same mechanism used to publish this document. Users may also wish to extend the set of fields that they use privately. Such "user-defined fields" are permitted.

The framework severely constrains document tone and appearance and is primarily useful for most intra-organization communications and well-structured inter-organization communication. It also can be used for some types of inter-process communication, such as simple file transfer and remote job entry. A more robust framework might allow for multi-font, multi-color, multi-dimension encoding of information. A less robust one, as is present in most single-machine message systems, would more severely constrain the ability to add fields and the decision to include specific fields. In contrast with paper-based communication, it is interesting to note that the RECEIVER of a message can exercise an extraordinary amount of control over the message's appearance. The amount of actual control available to message receivers is contingent upon the capabilities of their individual message systems.

2. NOTATIONAL CONVENTIONS

This specification uses an augmented Backus-Naur Form (BNF) notation. The differences from standard BNF involve naming rules and indicating repetition and "local" alternatives.

2.1. RULE NAMING

Angle brackets ("<", ">") are not used, in general. The name of a rule is simply the name itself, rather than "<name>". Quotation-marks enclose literal text (which may be upper and/or lower case). Certain basic rules are in uppercase, such as SPACE, TAB, CRLF, DIGIT, ALPHA, etc. Angle brackets are used in rule definitions, and in the rest of this document, whenever their presence will facilitate discerning the use of rule names.

2.2. RULE1 / RULE2: ALTERNATIVES

Elements separated by slash ("/") are alternatives. Therefore "foo / bar" will accept foo or bar.

2.3. (RULE1 RULE2): LOCAL ALTERNATIVES

Elements enclosed in parentheses are treated as a single element. Thus, "(elem (foo / bar) elem)" allows the token sequences "elem foo elem" and "elem bar elem".

2.4. *RULE: REPETITION

The character "*" preceding an element indicates repetition. The full form is:

```
<l>*<m>element
```

indicating at least <l> and at most <m> occurrences of element. Default values are 0 and infinity so that "*(element)" allows any number, including zero; "1*element" requires at least one; and "1*2element" allows one or two.

2.5. [RULE]: OPTIONAL

Square brackets enclose optional elements; "[foo bar]" is equivalent to "*1(foo bar)".

2.6. NRULE: SPECIFIC REPETITION

"<n>(element)" is equivalent to "<n>*<n>(element)"; that is, exactly <n> occurrences of (element). Thus 2DIGIT is a 2-digit number, and 3ALPHA is a string of three alphabetic characters.

2.7. #RULE: LISTS

A construct "#" is defined, similar to "*", as follows:

```
<l>#<m>element
```

indicating at least <l> and at most <m> elements, each separated by one or more commas (","). This makes the usual form of lists very easy; a rule such as '(element *("," element))' can be shown as "1#element". Wherever this construct is used, null elements are allowed, but do not contribute to the count of elements present. That is, "(element),,(element)" is permitted, but counts as only two elements. Therefore, where at least one element is required, at least one non-null element must be present.

Default values are 0 and infinity so that "#(element)" allows any number, including zero; "1#element" requires at least one; and "1#2element" allows one or two.

2.8. ; COMMENTS

A semi-colon, set off some distance to the right of rule text, starts a comment that continues to the end of line. This is a simple way of including useful notes in parallel with the specifications.

3. LEXICAL ANALYSIS OF MESSAGES

3.1. GENERAL DESCRIPTION

A message consists of header fields and, optionally, a body. The body is simply a sequence of lines containing ASCII characters. It is separated from the headers by a null line (i.e., a line with nothing preceding the CRLF).

3.1.1. LONG HEADER FIELDS

Each header field can be viewed as a single, logical line of ASCII characters, comprising a field-name and a field-body. For convenience, the field-body portion of this conceptual entity can be split into a multiple-line representation; this is called "folding". The general rule is that wherever there may be linear-white-space (NOT simply LWSP-chars), a CRLF immediately followed by AT LEAST one LWSP-char may instead be inserted. Thus, the single line

```
To:  "Joe & J. Harvey" <ddd @Org>, JJV @ BBN
```

can be represented as:

```
To:  "Joe & J. Harvey" <ddd @ Org>,
        JJV@BBN
```

and

```
To:  "Joe & J. Harvey"
                <ddd@ Org>, JJV
 @BBN
```

and

```
To:  "Joe &
 J. Harvey" <ddd @ Org>, JJV @ BBN
```

The process of moving from this folded multiple-line representation of a header field to its single line representation is called "unfolding". Unfolding is accomplished by regarding CRLF immediately followed by a LWSP-char as equivalent to the LWSP-char.

Note: While the standard permits folding wherever linear-white-space is permitted, it is recommended that structured fields, such as those containing addresses, limit folding to higher-level syntactic breaks. For address fields, it is recommended that such folding occur

between addresses, after the separating comma.

3.1.2. STRUCTURE OF HEADER FIELDS

Once a field has been unfolded, it may be viewed as being composed of a field-name followed by a colon (":"), followed by a field-body, and terminated by a carriage-return/line-feed. The field-name must be composed of printable ASCII characters (i.e., characters that have values between 33. and 126., decimal, except colon). The field-body may be composed of any ASCII characters, except CR or LF. (While CR and/or LF may be present in the actual text, they are removed by the action of unfolding the field.)

Certain field-bodies of headers may be interpreted according to an internal syntax that some systems may wish to parse. These fields are called "structured fields". Examples include fields containing dates and addresses. Other fields, such as "Subject" and "Comments", are regarded simply as strings of text.

Note: Any field which has a field-body that is defined as other than simply <text> is to be treated as a structured field.

Field-names, unstructured field bodies and structured field bodies each are scanned by their own, independent "lexical" analyzers.

3.1.3. UNSTRUCTURED FIELD BODIES

For some fields, such as "Subject" and "Comments", no structuring is assumed, and they are treated simply as <text>s, as in the message body. Rules of folding apply to these fields, so that such field bodies which occupy several lines must therefore have the second and successive lines indented by at least one LWSP-char.

3.1.4. STRUCTURED FIELD BODIES

To aid in the creation and reading of structured fields, the free insertion of linear-white-space (which permits folding by inclusion of CRLFs) is allowed between lexical tokens. Rather than obscuring the syntax specifications for these structured fields with explicit syntax for this linear-white-space, the existence of another "lexical" analyzer is assumed. This analyzer does not apply for unstructured field bodies that are simply strings of text, as described above. The analyzer provides an interpretation of the unfolded text

composing the body of the field as a sequence of lexical symbols.

These symbols are:

- individual special characters
- quoted-strings
- domain-literals
- comments
- atoms

The first four of these symbols are self-delimiting. Atoms are not; they are delimited by the self-delimiting symbols and by linear-white-space. For the purposes of regenerating sequences of atoms and quoted-strings, exactly one SPACE is assumed to exist, and should be used, between them. (Also, in the "Clarifications" section on "White Space", below, note the rules about treatment of multiple contiguous LWSP-chars.)

So, for example, the folded body of an address field

```
":sysmail"@  Some-Group. Some-Org,
Muhammed.(I am  the greatest) Ali @(the)Vegas.WBA
```

is analyzed into the following lexical symbols and types:

```
:sysmail               quoted string
@                      special
Some-Group             atom
.                      special
Some-Org               atom
,                      special
Muhammed               atom
.                      special
(I am  the greatest)   comment
Ali                    atom
@                      atom
(the)                  comment
Vegas                  atom
.                      special
WBA                    atom
```

The canonical representations for the data in these addresses are the following strings:

":sysmail"@Some-Group.Some-Org

and

Muhammed.Ali@Vegas.WBA

Note: For purposes of display, and when passing such structured information to other systems, such as mail protocol services, there must be NO linear-white-space between <word>s that are separated by period (".") or at-sign ("@") and exactly one SPACE between all other <word>s. Also, headers should be in a folded form.

3.2. HEADER FIELD DEFINITIONS

These rules show a field meta-syntax, without regard for the particular type or internal syntax. Their purpose is to permit detection of fields; also, they present to higher-level parsers an image of each field as fitting on one line.

```
field       =  field-name ":" [ field-body ] CRLF

field-name  =  1*<any CHAR, excluding CTLs, SPACE, and ":">

field-body  =  field-body-contents
               [CRLF LWSP-char field-body]

field-body-contents =
              <the ASCII characters making up the field-body, as
               defined in the following sections, and consisting
               of combinations of atom, quoted-string, and
               specials tokens, or else consisting of texts>
```

3.3. LEXICAL TOKENS

The following rules are used to define an underlying lexical analyzer, which feeds tokens to higher level parsers. See the ANSI references, in the Bibliography.

```
                                            ; (  Octal, Decimal.)
CHAR        =  <any ASCII character>        ; (  0-177,  0.-127.)
ALPHA       =  <any ASCII alphabetic character>
                                            ; (101-132, 65.- 90.)
                                            ; (141-172, 97.-122.)
DIGIT       =  <any ASCII decimal digit>    ; ( 60- 71, 48.- 57.)
CTL         =  <any ASCII control           ; (  0- 37,  0.- 31.)
                character and DEL>          ; (    177,     127.)
CR          =  <ASCII CR, carriage return>  ; (     15,      13.)
LF          =  <ASCII LF, linefeed>         ; (     12,      10.)
SPACE       =  <ASCII SP, space>            ; (     40,      32.)
HTAB        =  <ASCII HT, horizontal-tab>   ; (     11,       9.)
<">         =  <ASCII quote mark>           ; (     42,      34.)
CRLF        =  CR LF

LWSP-char   =  SPACE / HTAB                 ; semantics = SPACE

linear-white-space =  1*([CRLF] LWSP-char)  ; semantics = SPACE
                                            ; CRLF => folding

specials    =  "(" / ")" / "<" / ">" / "@"  ; Must be in quoted-
            /  "," / ";" / ":" / "\" / <">  ;  string, to use
            /  "." / "[" / "]"              ;  within a word.

delimiters  =  specials / linear-white-space / comment

text        =  <any CHAR, including bare    ; => atoms, specials,
                CR & bare LF, but NOT       ;  comments and
                including CRLF>             ;  quoted-strings are
```

```
                                            ;  NOT recognized.

atom        =  1*<any CHAR except specials, SPACE and CTLs>

quoted-string = <"> *(qtext/quoted-pair) <">; Regular qtext or
                                            ;   quoted chars.

qtext       =  <any CHAR excepting <">,     ; => may be folded
                "\" & CR, and including
                linear-white-space>

domain-literal =  "[" *(dtext / quoted-pair) "]"
```



```
dtext       =  <any CHAR excluding "[",     ; => may be folded
                "]", "\" & CR, & including
                linear-white-space>

comment     =  "(" *(ctext / quoted-pair / comment) ")"

ctext       =  <any CHAR excluding "(",     ; => may be folded
                ")", "\" & CR, & including
                linear-white-space>

quoted-pair =  "\" CHAR                     ; may quote any char

phrase      =  1*word                       ; Sequence of words

word        =  atom / quoted-string
```

3.4. CLARIFICATIONS

3.4.1. QUOTING

Some characters are reserved for special interpretation, such as delimiting lexical tokens. To permit use of these characters as uninterpreted data, a quoting mechanism is provided. To quote a character, precede it with a backslash ("\").

This mechanism is not fully general. Characters may be quoted only within a subset of the lexical constructs. In particular, quoting is limited to use within:

- quoted-string
- domain-literal
- comment

Within these constructs, quoting is REQUIRED for CR and "\" and for the character(s) that delimit the token (e.g., "(" and ")" for a comment). However, quoting is PERMITTED for any character.

Note: In particular, quoting is NOT permitted within atoms. For example when the local-part of an addr-spec must contain a special character, a quoted string must be used. Therefore, a specification such as:

Full\ Name@Domain

is not legal and must be specified as:

"Full Name"@Domain

3.4.2. WHITE SPACE

Note: In structured field bodies, multiple linear space ASCII characters (namely HTABs and SPACEs) are treated as single spaces and may freely surround any symbol. In all header fields, the only place in which at least one LWSP-char is REQUIRED is at the beginning of continuation lines in a folded field.

When passing text to processes that do not interpret text according to this standard (e.g., mail protocol servers), then NO linear-white-space characters should occur between a period (".") or at-sign ("@") and a <word>. Exactly ONE SPACE should be used in place of arbitrary linear-white-space and comment sequences.

Note: Within systems conforming to this standard, wherever a member of the list of delimiters is allowed, LWSP-chars may also occur before and/or after it.

Writers of mail-sending (i.e., header-generating) programs should realize that there is no network-wide definition of the effect of ASCII HT (horizontal-tab) characters on the appearance of text at another network host; therefore, the use of tabs in message headers, though permitted, is discouraged.

3.4.3. COMMENTS

A comment is a set of ASCII characters, which is enclosed in matching parentheses and which is not within a quoted-string The comment construct permits message originators to add text which will be useful for human readers, but which will be ignored by the formal semantics. Comments should be retained while the message is subject to interpretation according to this standard. However, comments must NOT be included in other cases, such as during protocol exchanges with mail servers.

Comments nest, so that if an unquoted left parenthesis occurs in a comment string, there must also be a matching right parenthesis. When a comment acts as the delimiter between a sequence of two lexical symbols, such as two atoms, it is lexically equivalent with a single SPACE, for the purposes of regenerating the sequence, such as when passing the sequence onto a mail protocol server. Comments are detected as such only within field-bodies of structured fields.

If a comment is to be "folded" onto multiple lines, then the syntax for folding must be adhered to. (See the "Lexical

Analysis of Messages" section on "Folding Long Header Fields" above, and the section on "Case Independence" below.) Note that the official semantics therefore do not "see" any unquoted CRLFs that are in comments, although particular parsing programs may wish to note their presence. For these programs, it would be reasonable to interpret a "CRLF LWSP-char" as being a CRLF that is part of the comment; i.e., the CRLF is kept and the LWSP-char is discarded. Quoted CRLFs (i.e., a backslash followed by a CR followed by a LF) still must be followed by at least one LWSP-char.

3.4.4. DELIMITING AND QUOTING CHARACTERS

The quote character (backslash) and characters that delimit syntactic units are not, generally, to be taken as data that are part of the delimited or quoted unit(s). In particular, the quotation-marks that define a quoted-string, the parentheses that define a comment and the backslash that quotes a following character are NOT part of the quoted-string, comment or quoted character. A quotation-mark that is to be part of a quoted-string, a parenthesis that is to be part of a comment and a backslash that is to be part of either must each be preceded by the quote-character backslash ("\"). Note that the syntax allows any character to be quoted within a quoted-string or comment; however only certain characters MUST be quoted to be included as data. These characters are the ones that are not part of the alternate text group (i.e., ctext or qtext).

The one exception to this rule is that a single SPACE is assumed to exist between contiguous words in a phrase, and this interpretation is independent of the actual number of LWSP-chars that the creator places between the words. To include more than one SPACE, the creator must make the LWSP-chars be part of a quoted-string.

Quotation marks that delimit a quoted string and backslashes that quote the following character should NOT accompany the quoted-string when the string is passed to processes that do not interpret data according to this specification (e.g., mail protocol servers).

3.4.5. QUOTED-STRINGS

Where permitted (i.e., in words in structured fields) quoted-strings are treated as a single symbol. That is, a quoted-string is equivalent to an atom, syntactically. If a quoted-string is to be "folded" onto multiple lines, then the syntax for folding must be adhered to. (See the "Lexical Analysis of

Messages" section on "Folding Long Header Fields" above, and the section on "Case Independence" below.) Therefore, the official semantics do not "see" any bare CRLFs that are in quoted-strings; however particular parsing programs may wish to note their presence. For such programs, it would be reasonable to interpret a "CRLF LWSP-char" as being a CRLF which is part of the quoted-string; i.e., the CRLF is kept and the LWSP-char is discarded. Quoted CRLFs (i.e., a backslash followed by a CR followed by a LF) are also subject to rules of folding, but the presence of the quoting character (backslash) explicitly indicates that the CRLF is data to the quoted string. Stripping off the first following LWSP-char is also appropriate when parsing quoted CRLFs.

3.4.6. BRACKETING CHARACTERS

There is one type of bracket which must occur in matched pairs and may have pairs nested within each other:

- o Parentheses ("(" and ")") are used to indicate comments.

There are three types of brackets which must occur in matched pairs, and which may NOT be nested:

- o Colon/semi-colon (":" and ";") are used in address specifications to indicate that the included list of addresses are to be treated as a group.

- o Angle brackets ("<" and ">") are generally used to indicate the presence of a one machine-usable reference (e.g., delimiting mailboxes), possibly including source-routing to the machine.

- o Square brackets ("[" and "]") are used to indicate the presence of a domain-literal, which the appropriate name-domain is to use directly, bypassing normal name-resolution mechanisms.

3.4.7. CASE INDEPENDENCE

Except as noted, alphabetic strings may be represented in any combination of upper and lower case. The only syntactic units which requires preservation of case information are:

- text
- qtext
- dtext
- ctext
- quoted-pair
- local-part, except "Postmaster"

When matching any other syntactic unit, case is to be ignored. For example, the field-names "From", "FROM", "from", and even "FroM" are semantically equal and should all be treated identically.

When generating these units, any mix of upper and lower case alphabetic characters may be used. The case shown in this specification is suggested for message-creating processes.

Note: The reserved local-part address unit, "Postmaster", is an exception. When the value "Postmaster" is being interpreted, it must be accepted in any mixture of case, including "POSTMASTER", and "postmaster".

3.4.8. FOLDING LONG HEADER FIELDS

Each header field may be represented on exactly one line consisting of the name of the field and its body, and terminated by a CRLF; this is what the parser sees. For readability, the field-body portion of long header fields may be "folded" onto multiple lines of the actual field. "Long" is commonly interpreted to mean greater than 65 or 72 characters. The former length serves as a limit, when the message is to be viewed on most simple terminals which use simple display software; however, the limit is not imposed by this standard.

Note: Some display software often can selectively fold lines, to suit the display terminal. In such cases, sender-provided folding can interfere with the display software.

3.4.9. BACKSPACE CHARACTERS

ASCII BS characters (Backspace, decimal 8) may be included in texts and quoted-strings to effect overstriking. However, any use of backspaces which effects an overstrike to the left of the beginning of the text or quoted-string is prohibited.

3.4.10. NETWORK-SPECIFIC TRANSFORMATIONS

During transmission through heterogeneous networks, it may be necessary to force data to conform to a network's local conventions. For example, it may be required that a CR be followed either by LF, making a CRLF, or by <null>, if the CR is to stand alone). Such transformations are reversed, when the message exits that network.

When crossing network boundaries, the message should be treated as passing through two modules. It will enter the first module containing whatever network-specific transformations that were necessary to permit migration through the "current" network. It then passes through the modules:

- o Transformation Reversal

  The "current" network's idiosyncracies are removed and the message is returned to the canonical form specified in this standard.

- o Transformation

  The "next" network's local idiosyncracies are imposed on the message.

```
                ------------------
    From   ==>  | Remove Net-A   |
    Net-A       | idiosyncracies |
                ------------------
                       ||
                       \/
                  Conformance
                  with standard
                       ||
                       \/
                ------------------
                | Impose Net-B   |  ==>  To
                | idiosyncracies |       Net-B
                ------------------
```

4. MESSAGE SPECIFICATION

4.1. SYNTAX

Note: Due to an artifact of the notational conventions, the syntax indicates that, when present, some fields, must be in a particular order. Header fields are NOT required to occur in any particular order, except that the message body must occur AFTER the headers. It is recommended that, if present, headers be sent in the order "Return-Path", "Received", "Date", "From", "Subject", "Sender", "To", "cc", etc.

This specification permits multiple occurrences of most fields. Except as noted, their interpretation is not specified here, and their use is discouraged.

The following syntax for the bodies of various fields should be thought of as describing each field body as a single long string (or line). The "Lexical Analysis of Message" section on "Long Header Fields", above, indicates how such long strings can be represented on more than one line in the actual transmitted message.

```
message     =  fields *( CRLF *text )       ; Everything after
                                            ;  first null line
                                            ;  is message body

fields      =    dates                      ; Creation time,
                 source                     ;  author id & one
               1*destination                ;  address required
                *optional-field             ;  others optional

source      = [  trace ]                    ; net traversals
                 originator                 ; original mail
              [  resent ]                   ; forwarded

trace       =    return                     ; path to sender
               1*received                   ; receipt tags

return      =  "Return-path" ":" route-addr ; return address

received    =  "Received"    ":"            ; one per relay
                  ["from" domain]           ; sending host
                  ["by"   domain]           ; receiving host
                  ["via"  atom]             ; physical path
                 *("with" atom)             ; link/mail protocol
                  ["id"   msg-id]           ; receiver msg id
                  ["for"  addr-spec]        ; initial form
```



```
                   ";"    date-time         ; time received
```

```
originator  =   authentic                   ; authenticated addr
              [ "Reply-To"   ":" 1#address] )

authentic   =   "From"       ":"   mailbox  ; Single author
            / ( "Sender"     ":"   mailbox  ; Actual submittor
                "From"       ":" 1#mailbox) ; Multiple authors
                                            ;  or not sender

resent      =   resent-authentic
              [ "Resent-Reply-To"  ":" 1#address] )

resent-authentic =
            =   "Resent-From"      ":"   mailbox
            / ( "Resent-Sender"    ":"   mailbox
                "Resent-From"      ":" 1#mailbox  )

dates       =   orig-date                   ; Original
              [ resent-date ]               ; Forwarded

orig-date   =  "Date"        ":"   date-time

resent-date =  "Resent-Date" ":"   date-time

destination =  "To"          ":" 1#address  ; Primary
            /  "Resent-To"   ":" 1#address
            /  "cc"          ":" 1#address  ; Secondary
            /  "Resent-cc"   ":" 1#address
            /  "bcc"         ":"  #address  ; Blind carbon
            /  "Resent-bcc"  ":"  #address

optional-field =
            /  "Message-ID"        ":"   msg-id
            /  "Resent-Message-ID" ":"   msg-id
            /  "In-Reply-To"       ":"  *(phrase / msg-id)
            /  "References"        ":"  *(phrase / msg-id)
            /  "Keywords"          ":"  #phrase
            /  "Subject"           ":"  *text
            /  "Comments"          ":"  *text
            /  "Encrypted"         ":" 1#2word
            /  extension-field              ; To be defined
            /  user-defined-field           ; May be pre-empted

msg-id      =  "<" addr-spec ">"            ; Unique message id
```



```
extension-field =
              <Any field which is defined in a document
               published as a formal extension to this
               specification; none will have names beginning
               with the string "X-">
```

```
user-defined-field =
             <Any field which has not been defined
              in this specification or published as an
              extension to this specification; names for
              such fields must be unique and may be
              pre-empted by published extensions>
```

4.2. FORWARDING

Some systems permit mail recipients to forward a message, retaining the original headers, by adding some new fields. This standard supports such a service, through the "Resent-" prefix to field names.

Whenever the string "Resent-" begins a field name, the field has the same semantics as a field whose name does not have the prefix. However, the message is assumed to have been forwarded by an original recipient who attached the "Resent-" field. This new field is treated as being more recent than the equivalent, original field. For example, the "Resent-From", indicates the person that forwarded the message, whereas the "From" field indicates the original author.

Use of such precedence information depends upon participants' communication needs. For example, this standard does not dictate when a "Resent-From:" address should receive replies, in lieu of sending them to the "From:" address.

Note: In general, the "Resent-" fields should be treated as containing a set of information that is independent of the set of original fields. Information for one set should not automatically be taken from the other. The interpretation of multiple "Resent-" fields, of the same type, is undefined.

In the remainder of this specification, occurrence of legal "Resent-" fields are treated identically with the occurrence of

fields whose names do not contain this prefix.

4.3. TRACE FIELDS

Trace information is used to provide an audit trail of message handling. In addition, it indicates a route back to the sender of the message.

The list of known "via" and "with" values are registered with the Network Information Center, SRI International, Menlo Park, California.

4.3.1. RETURN-PATH

This field is added by the final transport system that delivers the message to its recipient. The field is intended to contain definitive information about the address and route back to the message's originator.

Note: The "Reply-To" field is added by the originator and serves to direct replies, whereas the "Return-Path" field is used to identify a path back to the originator.

While the syntax indicates that a route specification is optional, every attempt should be made to provide that information in this field.

4.3.2. RECEIVED

A copy of this field is added by each transport service that relays the message. The information in the field can be quite useful for tracing transport problems.

The names of the sending and receiving hosts and time-of-receipt may be specified. The "via" parameter may be used, to indicate what physical mechanism the message was sent over, such as Arpanet or Phonenet, and the "with" parameter may be used to indicate the mail-, or connection-, level protocol that was used, such as the SMTP mail protocol, or X.25 transport protocol.

Note: Several "with" parameters may be included, to fully specify the set of protocols that were used.

Some transport services queue mail; the internal message identifier that is assigned to the message may be noted, using the "id" parameter. When the sending host uses a destination address specification that the receiving host reinterprets, by

expansion or transformation, the receiving host may wish to record the original specification, using the "for" parameter. For example, when a copy of mail is sent to the member of a distribution list, this parameter may be used to record the original address that was used to specify the list.

4.4. ORIGINATOR FIELDS

The standard allows only a subset of the combinations possible with the From, Sender, Reply-To, Resent-From, Resent-Sender, and Resent-Reply-To fields. The limitation is intentional.

4.4.1. FROM / RESENT-FROM

This field contains the identity of the person(s) who wished this message to be sent. The message-creation process should default this field to be a single, authenticated machine address, indicating the AGENT (person, system or process) entering the message. If this is not done, the "Sender" field MUST be present. If the "From" field IS defaulted this way, the "Sender" field is optional and is redundant with the "From" field. In all cases, addresses in the "From" field must be machine-usable (addr-specs) and may not contain named lists (groups).

4.4.2. SENDER / RESENT-SENDER

This field contains the authenticated identity of the AGENT (person, system or process) that sends the message. It is intended for use when the sender is not the author of the message, or to indicate who among a group of authors actually sent the message. If the contents of the "Sender" field would be completely redundant with the "From" field, then the "Sender" field need not be present and its use is discouraged (though still legal). In particular, the "Sender" field MUST be present if it is NOT the same as the "From" Field.

The Sender mailbox specification includes a word sequence which must correspond to a specific agent (i.e., a human user or a computer program) rather than a standard address. This indicates the expectation that the field will identify the single AGENT (person, system, or process) responsible for sending the mail and not simply include the name of a mailbox from which the mail was sent. For example in the case of a shared login name, the name, by itself, would not be adequate. The local-part address unit, which refers to this agent, is expected to be a computer system term, and not (for example) a generalized person reference which can be used outside the network text message context.

Since the critical function served by the "Sender" field is identification of the agent responsible for sending mail and since computer programs cannot be held accountable for their behavior, it is strongly recommended that when a computer program generates a message, the HUMAN who is responsible for that program be referenced as part of the "Sender" field mailbox specification.

4.4.3. REPLY-TO / RESENT-REPLY-TO

This field provides a general mechanism for indicating any mailbox(es) to which responses are to be sent. Three typical uses for this feature can be distinguished. In the first case, the author(s) may not have regular machine-based mailboxes and therefore wish(es) to indicate an alternate machine address. In the second case, an author may wish additional persons to be made aware of, or responsible for, replies. A somewhat different use may be of some help to "text message teleconferencing" groups equipped with automatic distribution services: include the address of that service in the "Reply-To" field of all messages submitted to the teleconference; then participants can "reply" to conference submissions to guarantee the correct distribution of any submission of their own.

Note: The "Return-Path" field is added by the mail transport service, at the time of final deliver. It is intended to identify a path back to the orginator of the message. The "Reply-To" field is added by the message originator and is intended to direct replies.

4.4.4. AUTOMATIC USE OF FROM / SENDER / REPLY-TO

For systems which automatically generate address lists for replies to messages, the following recommendations are made:

- o The "Sender" field mailbox should be sent notices of any problems in transport or delivery of the original messages. If there is no "Sender" field, then the "From" field mailbox should be used.

- o The "Sender" field mailbox should NEVER be used automatically, in a recipient's reply message.

- o If the "Reply-To" field exists, then the reply should go to the addresses indicated in that field and not to the address(es) indicated in the "From" field.

- o If there is a "From" field, but no "Reply-To" field, the reply should be sent to the address(es) indicated in the "From" field.

Sometimes, a recipient may actually wish to communicate with the person that initiated the message transfer. In such cases, it is reasonable to use the "Sender" address.

This recommendation is intended only for automated use of originator-fields and is not intended to suggest that replies may not also be sent to other recipients of messages. It is up to the respective mail-handling programs to decide what additional facilities will be provided.

Examples are provided in Appendix A.

4.5. RECEIVER FIELDS

4.5.1. TO / RESENT-TO

This field contains the identity of the primary recipients of the message.

4.5.2. CC / RESENT-CC

This field contains the identity of the secondary (informational) recipients of the message.

4.5.3. BCC / RESENT-BCC

This field contains the identity of additional recipients of the message. The contents of this field are not included in copies of the message sent to the primary and secondary recipients. Some systems may choose to include the text of the "Bcc" field only in the author(s)'s copy, while others may also include it in the text sent to all those indicated in the "Bcc" list.

4.6. REFERENCE FIELDS

4.6.1. MESSAGE-ID / RESENT-MESSAGE-ID

This field contains a unique identifier (the local-part address unit) which refers to THIS version of THIS message. The uniqueness of the message identifier is guaranteed by the host which generates it. This identifier is intended to be machine readable and not necessarily meaningful to humans. A message identifier pertains to exactly one instantiation of a particular message; subsequent revisions to the message should

each receive new message identifiers.

4.6.2. IN-REPLY-TO

The contents of this field identify previous correspondence which this message answers. Note that if message identifiers are used in this field, they must use the msg-id specification format.

4.6.3. REFERENCES

The contents of this field identify other correspondence which this message references. Note that if message identifiers are used, they must use the msg-id specification format.

4.6.4. KEYWORDS

This field contains keywords or phrases, separated by commas.

4.7. OTHER FIELDS

4.7.1. SUBJECT

This is intended to provide a summary, or indicate the nature, of the message.

4.7.2. COMMENTS

Permits adding text comments onto the message without disturbing the contents of the message's body.

4.7.3. ENCRYPTED

Sometimes, data encryption is used to increase the privacy of message contents. If the body of a message has been encrypted, to keep its contents private, the "Encrypted" field can be used to note the fact and to indicate the nature of the encryption. The first <word> parameter indicates the software used to encrypt the body, and the second, optional <word> is intended to aid the recipient in selecting the proper decryption key. This code word may be viewed as an index to a table of keys held by the recipient.

Note: Unfortunately, headers must contain envelope, as well as contents, information. Consequently, it is necessary that they remain unencrypted, so that mail transport services may access them. Since names, addresses, and "Subject" field contents may contain

sensitive information, this requirement limits total message privacy.

Names of encryption software are registered with the Network Information Center, SRI International, Menlo Park, California.

4.7.4. EXTENSION-FIELD

A limited number of common fields have been defined in this document. As network mail requirements dictate, additional fields may be standardized. To provide user-defined fields with a measure of safety, in name selection, such extension-fields will never have names that begin with the string "X-".

Names of Extension-fields are registered with the Network Information Center, SRI International, Menlo Park, California.

4.7.5. USER-DEFINED-FIELD

Individual users of network mail are free to define and use additional header fields. Such fields must have names which are not already used in the current specification or in any definitions of extension-fields, and the overall syntax of these user-defined-fields must conform to this specification's rules for delimiting and folding fields. Due to the extension-field publishing process, the name of a user-defined-field may be pre-empted

Note: The prefatory string "X-" will never be used in the names of Extension-fields. This provides user-defined fields with a protected set of names.

5. DATE AND TIME SPECIFICATION

5.1. SYNTAX

```
date-time   =  [ day "," ] date time        ; dd mm yy
                                            ;  hh:mm:ss zzz

day         =  "Mon"  / "Tue" /  "Wed"  / "Thu"
            /  "Fri"  / "Sat" /  "Sun"

date        =  1*2DIGIT month 2DIGIT        ; day month year
                                            ;  e.g. 20 Jun 82

month       =  "Jan"  /  "Feb" /  "Mar"  /  "Apr"
            /  "May"  /  "Jun" /  "Jul"  /  "Aug"
            /  "Sep"  /  "Oct" /  "Nov"  /  "Dec"

time        =  hour zone                    ; ANSI and Military

hour        =  2DIGIT ":" 2DIGIT [":" 2DIGIT]
                                            ; 00:00:00 - 23:59:59

zone        =  "UT"  / "GMT"                ; Universal Time
                                            ; North American : UT
            /  "EST" / "EDT"                ;  Eastern:  - 5/ - 4
            /  "CST" / "CDT"                ;  Central:  - 6/ - 5
            /  "MST" / "MDT"                ;  Mountain: - 7/ - 6
            /  "PST" / "PDT"                ;  Pacific:  - 8/ - 7
```

```
            /  1ALPHA                       ; Military: Z = UT;
                                            ;  A:-1; (J not used)
                                            ;  M:-12; N:+1; Y:+12
            / ( ("+" / "-") 4DIGIT )        ; Local differential
                                            ;  hours+min. (HHMM)
```

5.2. SEMANTICS

If included, day-of-week must be the day implied by the date specification.

Time zone may be indicated in several ways. "UT" is Universal Time (formerly called "Greenwich Mean Time"); "GMT" is permitted as a reference to Universal Time. The military standard uses a single character for each zone. "Z" is Universal Time. "A" indicates one hour earlier, and "M" indicates 12 hours earlier; "N" is one hour later, and "Y" is 12 hours later. The letter "J" is not used. The other remaining two forms are taken from ANSI standard X3.51-1975. One allows explicit indication of the amount of offset from UT; the other uses common 3-character strings for indicating time zones in North America.

6. ADDRESS SPECIFICATION

6.1. SYNTAX

```
address     =  mailbox                      ; one addressee
            /  group                        ; named list

group       =  phrase ":" [#mailbox] ";"

mailbox     =  addr-spec                    ; simple address
            /  phrase route-addr            ; name & addr-spec

route-addr  =  "<" [route] addr-spec ">"

route       =  1#("@" domain) ":"           ; path-relative

addr-spec   =  local-part "@" domain        ; global address

local-part  =  word *("." word)             ; uninterpreted
                                            ; case-preserved

domain      =  sub-domain *("." sub-domain)

sub-domain  =  domain-ref / domain-literal

domain-ref  =  atom                         ; symbolic reference
```

6.2. SEMANTICS

A mailbox receives mail. It is a conceptual entity which does not necessarily pertain to file storage. For example, some sites may choose to print mail on their line printer and deliver the output to the addressee's desk.

A mailbox specification comprises a person, system or process name reference, a domain-dependent string, and a name-domain reference. The name reference is optional and is usually used to indicate the human name of a recipient. The name-domain reference specifies a sequence of sub-domains. The domain-dependent string is uninterpreted, except by the final sub-domain; the rest of the mail service merely transmits it as a literal string.

6.2.1. DOMAINS

A name-domain is a set of registered (mail) names. A name-domain specification resolves to a subordinate name-domain specification or to a terminal domain-dependent string. Hence, domain specification is extensible, permitting any number of registration levels.

Name-domains model a global, logical, hierarchical addressing scheme. The model is logical, in that an address specification is related to name registration and is not necessarily tied to transmission path. The model's hierarchy is a directed graph, called an in-tree, such that there is a single path from the root of the tree to any node in the hierarchy. If more than one path actually exists, they are considered to be different addresses.

The root node is common to all addresses; consequently, it is not referenced. Its children constitute "top-level" name-domains. Usually, a service has access to its own full domain specification and to the names of all top-level name-domains.

The "top" of the domain addressing hierarchy -- a child of the root -- is indicated by the right-most field, in a domain specification. Its child is specified to the left, its child to the left, and so on.

Some groups provide formal registration services; these constitute name-domains that are independent logically of specific machines. In addition, networks and machines implicitly compose name-domains, since their membership usually is registered in name tables.

In the case of formal registration, an organization implements a (distributed) data base which provides an address-to-route mapping service for addresses of the form:

person@registry.organization

Note that "organization" is a logical entity, separate from any particular communication network.

A mechanism for accessing "organization" is universally available. That mechanism, in turn, seeks an instantiation of the registry; its location is not indicated in the address specification. It is assumed that the system which operates under the name "organization" knows how to find a subordinate registry. The registry will then use the "person" string to determine where to send the mail specification.

The latter, network-oriented case permits simple, direct, attachment-related address specification, such as:

user@host.network

Once the network is accessed, it is expected that a message will go directly to the host and that the host will resolve

the user name, placing the message in the user's mailbox.

6.2.2. ABBREVIATED DOMAIN SPECIFICATION

Since any number of levels is possible within the domain hierarchy, specification of a fully qualified address can become inconvenient. This standard permits abbreviated domain specification, in a special case:

> For the address of the sender, call the left-most sub-domain Level N. In a header address, if all of the sub-domains above (i.e., to the right of) Level N are the same as those of the sender, then they do not have to appear in the specification. Otherwise, the address must be fully qualified.
>
> This feature is subject to approval by local sub-domains. Individual sub-domains may require their member systems, which originate mail, to provide full domain specification only. When permitted, abbreviations may be present only while the message stays within the sub-domain of the sender.
>
> Use of this mechanism requires the sender's sub-domain to reserve the names of all top-level domains, so that full specifications can be distinguished from abbreviated specifications.

For example, if a sender's address is:

sender@registry-A.registry-1.organization-X

and one recipient's address is:

recipient@registry-B.registry-1.organization-X

and another's is:

recipient@registry-C.registry-2.organization-X

then ".registry-1.organization-X" need not be specified in the the message, but "registry-C.registry-2" DOES have to be specified. That is, the first two addresses may be abbreviated, but the third address must be fully specified.

When a message crosses a domain boundary, all addresses must be specified in the full format, ending with the top-level name-domain in the right-most field. It is the responsibility of mail forwarding services to ensure that addresses conform

with this requirement. In the case of abbreviated addresses, the relaying service must make the necessary expansions. It should be noted that it often is difficult for such a service to locate all occurrences of address abbreviations. For example, it will not be possible to find such abbreviations within the body of the message. The "Return-Path" field can aid recipients in recovering from these errors.

Note: When passing any portion of an addr-spec onto a process which does not interpret data according to this standard (e.g., mail protocol servers). There must be NO LWSP-chars preceding or following the at-sign or any delimiting period ("."), such as shown in the above examples, and only ONE SPACE between contiguous <word>s.

6.2.3. DOMAIN TERMS

A domain-ref must be THE official name of a registry, network, or host. It is a symbolic reference, within a name sub-domain. At times, it is necessary to bypass standard mechanisms for resolving such references, using more primitive information, such as a network host address rather than its associated host name.

To permit such references, this standard provides the domain-literal construct. Its contents must conform with the needs of the sub-domain in which it is interpreted.

Domain-literals which refer to domains within the ARPA Internet specify 32-bit Internet addresses, in four 8-bit fields noted in decimal, as described in Request for Comments #820, "Assigned Numbers." For example:

[10.0.3.19]

Note: THE USE OF DOMAIN-LITERALS IS STRONGLY DISCOURAGED. It is permitted only as a means of bypassing temporary system limitations, such as name tables which are not complete.

The names of "top-level" domains, and the names of domains under in the ARPA Internet, are registered with the Network Information Center, SRI International, Menlo Park, California.

6.2.4. DOMAIN-DEPENDENT LOCAL STRING

The local-part of an addr-spec in a mailbox specification (i.e., the host's name for the mailbox) is understood to be

whatever the receiving mail protocol server allows. For example, some systems do not understand mailbox references of the form "P. D. Q. Bach", but others do.

This specification treats periods (".") as lexical separators. Hence, their presence in local-parts which are not quoted-strings, is detected. However, such occurrences carry NO semantics. That is, if a local-part has periods within it, an address parser will divide the local-part into several tokens, but the sequence of tokens will be treated as one uninterpreted unit. The sequence will be re-assembled, when the address is passed outside of the system such as to a mail protocol service.

For example, the address:

First.Last@Registry.Org

is legal and does not require the local-part to be surrounded with quotation-marks. (However, "First Last" DOES require quoting.) The local-part of the address, when passed outside of the mail system, within the Registry.Org domain, is "First.Last", again without quotation marks.

6.2.5. BALANCING LOCAL-PART AND DOMAIN

In some cases, the boundary between local-part and domain can be flexible. The local-part may be a simple string, which is used for the final determination of the recipient's mailbox. All other levels of reference are, therefore, part of the domain.

For some systems, in the case of abbreviated reference to the local and subordinate sub-domains, it may be possible to specify only one reference within the domain part and place the other, subordinate name-domain references within the local-part. This would appear as:

mailbox.sub1.sub2@this-domain

Such a specification would be acceptable to address parsers which conform to RFC #733, but do not support this newer Internet standard. While contrary to the intent of this standard, the form is legal.

Also, some sub-domains have a specification syntax which does not conform to this standard. For example:

sub-net.mailbox@sub-domain.domain

uses a different parsing sequence for local-part than for domain.

Note: As a rule, the domain specification should contain fields which are encoded according to the syntax of this standard and which contain generally-standardized information. The local-part specification should contain only that portion of the address which deviates from the form or intention of the domain field.

6.2.6. MULTIPLE MAILBOXES

An individual may have several mailboxes and wish to receive mail at whatever mailbox is convenient for the sender to access. This standard does not provide a means of specifying "any member of" a list of mailboxes.

A set of individuals may wish to receive mail as a single unit (i.e., a distribution list). The <group> construct permits specification of such a list. Recipient mailboxes are specified within the bracketed part (":" - ";"). A copy of the transmitted message is to be sent to each mailbox listed. This standard does not permit recursive specification of groups within groups.

While a list must be named, it is not required that the contents of the list be included. In this case, the <address> serves only as an indication of group distribution and would appear in the form:

name:;

Some mail services may provide a group-list distribution facility, accepting a single mailbox reference, expanding it to the full distribution list, and relaying the mail to the list's members. This standard provides no additional syntax for indicating such a service. Using the <group> address alternative, while listing one mailbox in it, can mean either that the mailbox reference will be expanded to a list or that there is a group with one member.

6.2.7. EXPLICIT PATH SPECIFICATION

At times, a message originator may wish to indicate the transmission path that a message should follow. This is called source routing. The normal addressing scheme, used in an addr-spec, is carefully separated from such information; the <route> portion of a route-addr is provided for such occasions. It specifies the sequence of hosts and/or transmission

services that are to be traversed. Both domain-refs and domain-literals may be used.

Note: The use of source routing is discouraged. Unless the sender has special need of path restriction, the choice of transmission route should be left to the mail transport service.

6.3. RESERVED ADDRESS

It often is necessary to send mail to a site, without knowing any of its valid addresses. For example, there may be mail system dysfunctions, or a user may wish to find out a person's correct address, at that site.

This standard specifies a single, reserved mailbox address (local-part) which is to be valid at each site. Mail sent to that address is to be routed to a person responsible for the site's mail system or to a person with responsibility for general site operation. The name of the reserved local-part address is:

Postmaster

so that "Postmaster@domain" is required to be valid.

Note: This reserved local-part must be matched without sensitivity to alphabetic case, so that "POSTMASTER", "postmaster", and even "poStmASteR" is to be accepted.

7. BIBLIOGRAPHY

ANSI. "USA Standard Code for Information Interchange," X3.4. American National Standards Institute: New York (1968). Also in: Feinler, E. and J. Postel, eds., "ARPANET Protocol Handbook", NIC 7104.

ANSI. "Representations of Universal Time, Local Time Differentials, and United States Time Zone References for Information Interchange," X3.51-1975. American National Standards Institute: New York (1975).

Bemer, R.W., "Time and the Computer." In: Interface Age (Feb. 1979).

Bennett, C.J. "JNT Mail Protocol". Joint Network Team, Rutherford and Appleton Laboratory: Didcot, England.

Bhushan, A.K., Pogran, K.T., Tomlinson, R.S., and White, J.E. "Standardizing Network Mail Headers," ARPANET Request for Comments No. 561, Network Information Center No. 18516; SRI International: Menlo Park (September 1973).

Birrell, A.D., Levin, R., Needham, R.M., and Schroeder, M.D. "Grapevine: An Exercise in Distributed Computing," Communications of the ACM 25, 4 (April 1982), 260-274.

Crocker, D.H., Vittal, J.J., Pogran, K.T., Henderson, D.A. "Standard for the Format of ARPA Network Text Message," ARPANET Request for Comments No. 733, Network Information Center No. 41952. SRI International: Menlo Park (November 1977).

Feinler, E.J. and Postel, J.B. ARPANET Protocol Handbook, Network Information Center No. 7104 (NTIS AD A003890). SRI International: Menlo Park (April 1976).

Harary, F. "Graph Theory". Addison-Wesley: Reading, Mass. (1969).

Levin, R. and Schroeder, M. "Transport of Electronic Messages through a Network," TeleInformatics 79, pp. 29-33. North Holland (1979). Also as Xerox Palo Alto Research Center Technical Report CSL-79-4.

Myer, T.H. and Henderson, D.A. "Message Transmission Protocol," ARPANET Request for Comments, No. 680, Network Information Center No. 32116. SRI International: Menlo Park (1975).

NBS. "Specification of Message Format for Computer Based Message Systems, Recommended Federal Information Processing Standard." National Bureau of Standards: Gaithersburg, Maryland (October 1981).

NIC. Internet Protocol Transition Workbook. Network Information Center, SRI-International, Menlo Park, California (March 1982).

Oppen, D.C. and Dalal, Y.K. "The Clearinghouse: A Decentralized Agent for Locating Named Objects in a Distributed Environment," OPD-T8103. Xerox Office Products Division: Palo Alto, CA. (October 1981).

Postel, J.B. "Assigned Numbers," ARPANET Request for Comments, No. 820. SRI International: Menlo Park (August 1982).

Postel, J.B. "Simple Mail Transfer Protocol," ARPANET Request for Comments, No. 821. SRI International: Menlo Park (August 1982).

Shoch, J.F. "Internetwork naming, addressing and routing," in Proc. 17th IEEE Computer Society International Conference, pp. 72-79, Sept. 1978, IEEE Cat. No. 78 CH 1388-8C.

Su, Z. and Postel, J. "The Domain Naming Convention for Internet User Applications," ARPANET Request for Comments, No. 819. SRI International: Menlo Park (August 1982).

APPENDIX

A. EXAMPLES

A.1. ADDRESSES

A.1.1. Alfred Neuman <Neuman@BBN-TENEXA>

A.1.2. Neuman@BBN-TENEXA

These two "Alfred Neuman" examples have identical semantics, as far as the operation of the local host's mail sending (distribution) program (also sometimes called its "mailer") and the remote host's mail protocol server are concerned. In the first example, the "Alfred Neuman" is ignored by the mailer, as "Neuman@BBN-TENEXA" completely specifies the recipient. The second example contains no superfluous information, and, again, "Neuman@BBN-TENEXA" is the intended recipient.

Note: When the message crosses name-domain boundaries, then these specifications must be changed, so as to indicate the remainder of the hierarchy, starting with the top level.

A.1.3. "George, Ted" <Shared@Group.Arpanet>

This form might be used to indicate that a single mailbox is shared by several users. The quoted string is ignored by the originating host's mailer, because "Shared@Group.Arpanet" completely specifies the destination mailbox.

A.1.4. Wilt . (the Stilt) Chamberlain@NBA.US

The "(the Stilt)" is a comment, which is NOT included in the destination mailbox address handed to the originating system's mailer. The local-part of the address is the string "Wilt.Chamberlain", with NO space between the first and second words.

A.1.5. Address Lists

```
Gourmets:  Pompous Person <WhoZiWhatZit@Cordon-Bleu>,
           Childs@WGBH.Boston, Galloping Gourmet@
           ANT.Down-Under (Australian National Television),
           Cheapie@Discount-Liquors;,
  Cruisers:  Port@Portugal, Jones@SEA;,
    Another@Somewhere.SomeOrg
```


This group list example points out the use of comments and the mixing of addresses and groups.

A.2. ORIGINATOR ITEMS

A.2.1. Author-sent

George Jones logs into his host as "Jones". He sends mail himself.

```
From:  Jones@Group.Org
```

or

```
From:  George Jones <Jones@Group.Org>
```

A.2.2. Secretary-sent

George Jones logs in as Jones on his host. His secretary, who logs in as Secy sends mail for him. Replies to the mail should go to George.

```
From:    George Jones <Jones@Group>
Sender:  Secy@Other-Group
```

A.2.3. Secretary-sent, for user of shared directory

George Jones' secretary sends mail for George. Replies should go to George.

```
From:     George Jones<Shared@Group.Org>
Sender:   Secy@Other-Group
```

Note that there need not be a space between "Jones" and the "<", but adding a space enhances readability (as is the case in other examples.

A.2.4. Committee activity, with one author

George is a member of a committee. He wishes to have any replies to his message go to all committee members.

```
From:     George Jones <Jones@Host.Net>
Sender:   Jones@Host
Reply-To: The Committee: Jones@Host.Net,
                         Smith@Other.Org,
                         Doe@Somewhere-Else;
```

Note that if George had not included himself in the

enumeration of The Committee, he would not have gotten an implicit reply; the presence of the "Reply-to" field SUPERSEDES the sending of a reply to the person named in the "From" field.

A.2.5. Secretary acting as full agent of author

George Jones asks his secretary (Secy@Host) to send a message for him in his capacity as Group. He wants his secretary to handle all replies.

```
From:     George Jones <Group@Host>
Sender:   Secy@Host
Reply-To: Secy@Host
```

A.2.6. Agent for user without online mailbox

A friend of George's, Sarah, is visiting. George's secretary sends some mail to a friend of Sarah in computer-land. Replies should go to George, whose mailbox is Jones at Registry.

```
From:     Sarah Friendly <Secy@Registry>
Sender:   Secy-Name <Secy@Registry>
Reply-To: Jones@Registry.
```

A.2.7. Agent for member of a committee

George's secretary sends out a message which was authored jointly by all the members of a committee. Note that the name of the committee cannot be specified, since <group> names are not permitted in the From field.

```
From:   Jones@Host,
        Smith@Other-Host,
        Doe@Somewhere-Else
Sender: Secy@SHost
```


A.3. COMPLETE HEADERS

A.3.1. Minimum required

```
Date:     26 Aug 76 1429 EDT        Date:     26 Aug 76 1429 EDT
From:     Jones@Registry.Org   or   From:     Jones@Registry.Org
Bcc:                                To:       Smith@Registry.Org
```

Note that the "Bcc" field may be empty, while the "To" field is required to have at least one address.

A.3.2. Using some of the additional fields

```
Date:     26 Aug 76 1430 EDT
From:     George Jones<Group@Host>
Sender:   Secy@SHOST
To:       "Al Neuman"@Mad-Host,
          Sam.Irving@Other-Host
Message-ID:  <some.string@SHOST>
```

A.3.3. About as complex as you're going to get

```
Date     :  27 Aug 76 0932 PDT
From     :  Ken Davis <KDavis@This-Host.This-net>
Subject  :  Re: The Syntax in the RFC
Sender   :  KSecy@Other-Host
Reply-To :  Sam.Irving@Reg.Organization
To       :  George Jones <Group@Some-Reg.An-Org>,
            Al.Neuman@MAD.Publisher
cc       :  Important folk:
              Tom Softwood <Balsa@Tree.Root>,
              "Sam Irving"@Other-Host;,
            Standard Distribution:
              /main/davis/people/standard@Other-Host,
              "<Jones>standard.dist.3"@Tops-20-Host>;
Comment  :  Sam is away on business. He asked me to handle
            his mail for him.  He'll be able to provide  a
            more  accurate  explanation  when  he  returns
            next week.
In-Reply-To: <some.string@DBM.Group>, George's message
X-Special-action:  This is a sample of user-defined field-
            names.  There could also be a field-name
            "Special-action", but its name might later be
            preempted
Message-ID: <4231.629.XYzi-What@Other-Host>
```


B. SIMPLE FIELD PARSING

Some mail-reading software systems may wish to perform only minimal processing, ignoring the internal syntax of structured field-bodies and treating them the same as unstructured-field-bodies. Such software will need only to distinguish:

- o Header fields from the message body,
- o Beginnings of fields from lines which continue fields,
- o Field-names from field-contents.

The abbreviated set of syntactic rules which follows will suffice for this purpose. It describes a limited view of messages and is a subset of the syntactic rules provided in the main part of this specification. One small exception is that the contents of field-bodies consist only of text:

B.1. SYNTAX

```
message         =   *field *(CRLF *text)

field           =    field-name ":" [field-body] CRLF

field-name      =  1*<any CHAR, excluding CTLs, SPACE, and ":">

field-body      =   *text [CRLF LWSP-char field-body]
```

B.2. SEMANTICS

Headers occur before the message body and are terminated by a null line (i.e., two contiguous CRLFs).

A line which continues a header field begins with a SPACE or HTAB character, while a line beginning a field starts with a printable character which is not a colon.

A field-name consists of one or more printable characters (excluding colon, space, and control-characters). A field-name MUST be contained on one line. Upper and lower case are not distinguished when comparing field-names.

C. DIFFERENCES FROM RFC #733

The following summarizes the differences between this standard and the one specified in Arpanet Request for Comments #733, "Standard for the Format of ARPA Network Text Messages". The differences are listed in the order of their occurrence in the current specification.

C.1. FIELD DEFINITIONS

C.1.1. FIELD NAMES

These now must be a sequence of printable characters. They may not contain any LWSP-chars.

C.2. LEXICAL TOKENS

C.2.1. SPECIALS

The characters period ("."), left-square bracket ("["), and right-square bracket ("]") have been added. For presentation purposes, and when passing a specification to a system that does not conform to this standard, periods are to be contiguous with their surrounding lexical tokens. No linear-white-space is permitted between them. The presence of one LWSP-char between other tokens is still directed.

C.2.2. ATOM

Atoms may not contain SPACE.

C.2.3. SPECIAL TEXT

ctext and qtext have had backslash ("\") added to the list of prohibited characters.

C.2.4. DOMAINS

The lexical tokens <domain-literal> and <dtext> have been added.

C.3. MESSAGE SPECIFICATION

C.3.1. TRACE

The "Return-path:" and "Received:" fields have been specified.

C.3.2. FROM

The "From" field must contain machine-usable addresses (addr-spec). Multiple addresses may be specified, but named-lists (groups) may not.

C.3.3. RESENT

The meta-construct of prefacing field names with the string "Resent-" has been added, to indicate that a message has been forwarded by an intermediate recipient.

C.3.4. DESTINATION

A message must contain at least one destination address field. "To" and "CC" are required to contain at least one address.

C.3.5. IN-REPLY-TO

The field-body is no longer a comma-separated list, although a sequence is still permitted.

C.3.6. REFERENCE

The field-body is no longer a comma-separated list, although a sequence is still permitted.

C.3.7. ENCRYPTED

A field has been specified that permits senders to indicate that the body of a message has been encrypted.

C.3.8. EXTENSION-FIELD

Extension fields are prohibited from beginning with the characters "X-".

C.4. DATE AND TIME SPECIFICATION

C.4.1. SIMPLIFICATION

Fewer optional forms are permitted and the list of three-letter time zones has been shortened.

C.5. ADDRESS SPECIFICATION

C.5.1. ADDRESS

The use of quoted-string, and the ":"-atom-":" construct, have been removed. An address now is either a single mailbox reference or is a named list of addresses. The latter indicates a group distribution.

C.5.2. GROUPS

Group lists are now required to to have a name. Group lists may not be nested.

C.5.3. MAILBOX

A mailbox specification may indicate a person's name, as before. Such a named list no longer may specify multiple mailboxes and may not be nested.

C.5.4. ROUTE ADDRESSING

Addresses now are taken to be absolute, global specifications, independent of transmission paths. The <route> construct has been provided, to permit explicit specification of transmission path. RFC #733's use of multiple at-signs ("@") was intended as a general syntax for indicating routing and/or hierarchical addressing. The current standard separates these specifications and only one at-sign is permitted.

C.5.5. AT-SIGN

The string " at " no longer is used as an address delimiter. Only at-sign ("@") serves the function.

C.5.6. DOMAINS

Hierarchical, logical name-domains have been added.

C.6. RESERVED ADDRESS

The local-part "Postmaster" has been reserved, so that users can be guaranteed at least one valid address at a site.

D. ALPHABETICAL LISTING OF SYNTAX RULES

```
address     =  mailbox                      ; one addressee
            /  group                        ; named list
addr-spec   =  local-part "@" domain        ; global address
ALPHA       =  <any ASCII alphabetic character>
                                            ; (101-132, 65.- 90.)
                                            ; (141-172, 97.-122.)
atom        =  1*<any CHAR except specials, SPACE and CTLs>
authentic   =   "From"       ":"   mailbox  ; Single author
            / ( "Sender"     ":"   mailbox  ; Actual submittor
                "From"       ":" 1#mailbox) ; Multiple authors
                                            ;  or not sender
CHAR        =  <any ASCII character>        ; (  0-177,  0.-127.)
comment     =  "(" *(ctext / quoted-pair / comment) ")"
CR          =  <ASCII CR, carriage return>  ; (     15,      13.)
CRLF        =  CR LF
ctext       =  <any CHAR excluding "(",     ; => may be folded
                ")", "\" & CR, & including
                linear-white-space>
CTL         =  <any ASCII control           ; (  0- 37,  0.- 31.)
                character and DEL>          ; (    177,     127.)
date        =  1*2DIGIT month 2DIGIT        ; day month year
                                            ;  e.g. 20 Jun 82
dates       =   orig-date                   ; Original
              [ resent-date ]               ; Forwarded
```

```
date-time   =  [ day "," ] date time        ; dd mm yy
                                            ;  hh:mm:ss zzz
day         =  "Mon"  / "Tue" /  "Wed"  / "Thu"
            /  "Fri"  / "Sat" /  "Sun"
delimiters  =  specials / linear-white-space / comment
destination =  "To"          ":" 1#address  ; Primary
            /  "Resent-To"   ":" 1#address
            /  "cc"          ":" 1#address  ; Secondary
            /  "Resent-cc"   ":" 1#address
            /  "bcc"         ":"  #address  ; Blind carbon
            /  "Resent-bcc"  ":"  #address
DIGIT       =  <any ASCII decimal digit>    ; ( 60- 71, 48.- 57.)
domain      =  sub-domain *("." sub-domain)
domain-literal =  "[" *(dtext / quoted-pair) "]"
domain-ref  =  atom                         ; symbolic reference
dtext       =  <any CHAR excluding "[",     ; => may be folded
                "]", "\" & CR, & including
                linear-white-space>
extension-field =
              <Any field which is defined in a document
               published as a formal extension to this
               specification; none will have names beginning
               with the string "X-">
```



```
field       =  field-name ":" [ field-body ] CRLF
fields      =    dates                      ; Creation time,
                 source                     ;  author id & one
               1*destination                ;  address required
                *optional-field             ;  others optional
field-body  =  field-body-contents
               [CRLF LWSP-char field-body]
field-body-contents =
              <the ASCII characters making up the field-body, as
               defined in the following sections, and consisting
               of combinations of atom, quoted-string, and
               specials tokens, or else consisting of texts>
field-name  =  1*<any CHAR, excluding CTLs, SPACE, and ":">
group       =  phrase ":" [#mailbox] ";"
hour        =  2DIGIT ":" 2DIGIT [":" 2DIGIT]
                                            ; 00:00:00 - 23:59:59
HTAB        =  <ASCII HT, horizontal-tab>   ; (     11,       9.)
LF          =  <ASCII LF, linefeed>         ; (     12,      10.)
linear-white-space =  1*([CRLF] LWSP-char)  ; semantics = SPACE
                                            ; CRLF => folding
local-part  =  word *("." word)             ; uninterpreted
                                            ; case-preserved
LWSP-char   =  SPACE / HTAB                 ; semantics = SPACE
mailbox     =  addr-spec                    ; simple address
            /  phrase route-addr            ; name & addr-spec
message     =  fields *( CRLF *text )       ; Everything after
                                            ;  first null line
                                            ;  is message body
month       =  "Jan"  /  "Feb" /  "Mar"  /  "Apr"
```

```
            /  "May"  /  "Jun" /  "Jul"  /  "Aug"
            /  "Sep"  /  "Oct" /  "Nov"  /  "Dec"
msg-id      =  "<" addr-spec ">"            ; Unique message id
optional-field =
            /  "Message-ID"        ":"   msg-id
            /  "Resent-Message-ID" ":"   msg-id
            /  "In-Reply-To"       ":"  *(phrase / msg-id)
            /  "References"        ":"  *(phrase / msg-id)
            /  "Keywords"          ":"  #phrase
            /  "Subject"           ":"  *text
            /  "Comments"          ":"  *text
            /  "Encrypted"         ":" 1#2word
            /  extension-field              ; To be defined
            /  user-defined-field           ; May be pre-empted
orig-date   =  "Date"        ":"   date-time
originator  =   authentic                   ; authenticated addr
              [ "Reply-To"   ":" 1#address] )
phrase      =  1*word                       ; Sequence of words
```



```
qtext       =  <any CHAR excepting <">,     ; => may be folded
                "\" & CR, and including
                linear-white-space>
quoted-pair =  "\" CHAR                     ; may quote any char
quoted-string = <"> *(qtext/quoted-pair) <">; Regular qtext or
                                            ;   quoted chars.
received    =  "Received"    ":"            ; one per relay
                  ["from" domain]           ; sending host
                  ["by"   domain]           ; receiving host
                  ["via"  atom]             ; physical path
                 *("with" atom)             ; link/mail protocol
                  ["id"   msg-id]           ; receiver msg id
                  ["for"  addr-spec]        ; initial form
                   ";"    date-time         ; time received

resent      =   resent-authentic
              [ "Resent-Reply-To"  ":" 1#address] )
resent-authentic =
            =   "Resent-From"      ":"   mailbox
            / ( "Resent-Sender"    ":"   mailbox
                "Resent-From"      ":" 1#mailbox  )
resent-date =  "Resent-Date" ":"   date-time
return      =  "Return-path" ":" route-addr ; return address
route       =  1#("@" domain) ":"           ; path-relative
route-addr  =  "<" [route] addr-spec ">"
source      = [  trace ]                    ; net traversals
                 originator                 ; original mail
              [  resent ]                   ; forwarded
SPACE       =  <ASCII SP, space>            ; (     40,      32.)
specials    =  "(" / ")" / "<" / ">" / "@"  ; Must be in quoted-
            /  "," / ";" / ":" / "\" / <">  ;  string, to use
            /  "." / "[" / "]"              ;  within a word.
```

```
sub-domain  =  domain-ref / domain-literal
text        =  <any CHAR, including bare    ; => atoms, specials,
                CR & bare LF, but NOT       ;  comments and
                including CRLF>             ;  quoted-strings are
                                            ;  NOT recognized.
time        =  hour zone                    ; ANSI and Military
trace       =    return                     ; path to sender
               1*received                   ; receipt tags
user-defined-field =
              <Any field which has not been defined
               in this specification or published as an
               extension to this specification; names for
               such fields must be unique and may be
               pre-empted by published extensions>
word        =  atom / quoted-string
```



```
zone        =  "UT"  / "GMT"                ; Universal Time
                                            ; North American : UT
            /  "EST" / "EDT"                ;  Eastern:  - 5/ - 4
            /  "CST" / "CDT"                ;  Central:  - 6/ - 5
            /  "MST" / "MDT"                ;  Mountain: - 7/ - 6
            /  "PST" / "PDT"                ;  Pacific:  - 8/ - 7
            /  1ALPHA                       ; Military: Z = UT;
<">         =  <ASCII quote mark>           ; (     42,      34.)
```