News Article Format

draft-ietf-usefor-article-01

USEFOR Working Group

Status of this Memo

It is hoped that this document will obsolete RFC 1036 and will become an Internet standard.

This document is a successor to Henry Spencer's "Son of 1036" Draft, and has been referred to as "Grandson of 1036".

Distribution of this memo is unlimited.

Abstract

This Draft defines the format of network news articles, and defines roles and responsibilities for humans and software.

Network news articles resemble mail messages but are broadcast to potentially large audiences, using a flooding algorithm that propagates one copy to each interested host (or group thereof), typically stores only one copy per host, and does not require any central administration or systematic registration of interested users. Network news originated as the medium of communication for Usenet, circa 1980.

The term "Usenet" refers to the protocols established in RFC 1036 and successors; the software implementing those protocols; the network of hosts exchanging traffic using that software; and also the traffic itself. Cooperating subnets are possible; these are groups of hosts which agree to hold each other and themselves to an internally adopted set of standards concerning protocol details or implementations. When a cooperating subnet does not exchange traffic with general Usenet hosts, then it is no longer a part of Usenet, but a separate entity.

Since then Usenet has grown explosively, and most Internet sites participate in it. In addition, the news technology is now in widespread use for other purposes, on the Internet and elsewhere.

This document is intended to provide a definitive guide to the article format and interpretations thereof. Backward compatibility is a major goal, but where this document and earlier documents or practices collide, this document should be used.

Table of Contents

1. Introduction

Network news articles resemble mail messages but are broadcast to potentially-large audiences, using a flooding algorithm that propagates one copy to each interested host (or groups thereof), typically stores only one copy per host, and does not require any central administration or systematic registration of interested users. Network news originated as the medium of communication for Usenet, circa 1980. Since then Usenet has grown explosively, and many Internet sites participate in it. In addition, the news technology is now in widespread use for other purposes, on the Internet and elsewhere.

The earliest news interchange used the so-called "A News" article format. Shortly thereafter, an article format vaguely resembling Internet mail was devised and used briefly. Both of those formats are completely obsolete; they are documented in appendix A for historical reasons only. With publication of RFC 850 [rrr] in 1983, news articles came to closely resemble Internet mail messages, with some restrictions and some additional headers. RFC 1036 in 1987 updated RFC 850 without making major changes.

A Draft popularly referred to as "Son of 1036" was written in 1992 by Henry Spencer. That document formed the original basis for this document. Much is taken directly from Son of 1036, and it is hoped that we have followed its spirit and intentions.

As in this document's predecessors, the exact means used to transmit articles from one host to another is not specified. NNTP [rrr] is the most common transmission method on the Internet, but a number of others are in use, including the UUCP protocol [rrr] extensively used in the early days of Usenet, FTP, tape archives, and physically delivered magnetic and optical media.

Several of the mechanisms described in this document may seem somewhat strange or even bizarre at first reading. As with Internet mail, there is no reasonable possibility of updating the entire installed base of news software promptly, so interoperability with old software is critical and will remain so. Compatibility with existing practice and robustness in an imperfect world necessarily take priority over elegance. Elegance is left to the implementors.

2. Definitions, Notations and Conventions

2.1 Definitions.

An "article" is the unit of news, analogous to a [MAIL] "message".

A "poster" is the person or software that composes and submits a possibly compliant article to an injecting agent. The poster is analogous to [MAIL]'s author(s).

A "posting agent" is software that assists posters to prepare articles, including adding required headers and determining whether the final article is compliant to this standard. If the article is compliant it passes the article on to an injecting agent for final checking and injection into the news stream. If the article is not compliant or rejected by the injecting agent then the posting agent informs the poster with an explanation of the error.

An "injecting agent" takes the finished article from the posting agent (often via the NNTP "post" command) performs some final checks and passes it on to a relaying agent for general distribution.

A "relaying agent" is software which receives allegedly compliant articles from injecting agents and/or other relaying agents, and possibly passes copies on to other relaying agents and serving agents.

A "serving agent" takes an article from a relaying agent and files it in a "news database" . It also provides an interface for reading agents to access the news database.

A "reader" is the person or software reading news articles.

A "reading agent" is software which presents articles to a reader.

A "newsgroup" is a single news forum, a logical bulletin board, having a name and nominally intended for articles on a specific topic. An article is "posted to" a single newsgroup or several newsgroups. When an article is posted to more than one newsgroup, it is said to be "crossposted"; note that this differs from posting the same text as part of each of several articles, one per newsgroup. A "hierarchy" is the set of all newsgroups whose names share a first component.

A newsgroup may be "moderated", in which case submissions are not posted directly, but mailed to a "moderator" for consideration and possible posting. Moderators are typically human but may be implemented partially or entirely in software.

A "followup" is an article containing a response to the contents of an earlier article (the followup's "precursor").

A "followup agent" is a combination of reading agent, and posting agent that aids in the preparation and posting of a followup.

A "reply agent" is a combination of reading agent and mailer that aids in the preparation and posting of an email response to an article.

A "message ID" is a unique identifier for an article, usually supplied by the posting agent which posted it. It distinguishes the article from every other article ever posted anywhere. Articles with the same message ID are treated as identical copies of the same article even if they are not in fact identical.

A "gateway" is software which receives news articles and converts them to messages of some other kind (e.g. mail to a mailing list), or vice versa; in essence it is a translating relaying agent that straddles boundaries between different methods of message exchange. The most common type of gateway connects newsgroup(s) to mailing list(s), either unidirectionally or bidirectionally, but there are also gateways between news networks using this document's news format and those using other formats.

A "control message" is an article which is marked as containing control information; a relaying or serving agent receiving such an article may (subject to permissions etc.) take actions beyond just filing and passing on the article.

An article's "reply address" is the address to which mailed replies should be sent. This is the address specified in the article's From header (see section 5.2), unless it also has a Reply-To header (see section 6.3).

2.2. Textual Notations

Throughout this document, [MAIL] is short for "the current RFCs governing electronic mail formats, beginning with the historical RFC 822 and continuing to its modern successors.

"ASCII" is short for "the ANSI X3.4 character set" [rrr]. While "ASCII" is often misused to refer to various character sets somewhat similar to X3.4, in this document, "ASCII" means X3.4 and only X3.4. ASCII is a 7 bit character set. Please note that this document requires that all agents be 8 bit clean; that is, they must accept and transmit data without changing or omitting the 8th bit.

Certain words used to define the significance of individual requirements are capitalized. "MUST", "SHOULD", "MAY" and the same words followed by "NOT" should be read as having the same meaning as in RFC 2119.

This document contains explanatory notes using the following format. These may be skipped by persons interested solely in the content of the specification. The purpose of the notes is to explain why choices were made, to place them in context, or to suggest possible implementation techniques.

NOTE: While such explanatory notes may seem superfluous in principle, they often help the less-than-omniscient reader grasp the purpose of the specification and the constraints involved. Given the limitations of natural language for descriptive purposes, this improves the probability that implementors and users will understand the true intent of the specification in cases where the wording is not entirely clear.

All numeric values are given in decimal unless otherwise indicated. Octets are assumed to be unsigned values for this purpose.

Through this document we will give examples of various definitions, headers and other specifications. It MUST be remembered that these samples are for the aid of the reader only and do NOT define any specification themselves. In order to prevent possible conflict with "Real World" entities and people the top level domain of ".example" is used in all sample domains and addresses. The hierarchy of example.* is also used a sample hierarchy. Information on the ".example" top level domain is in [TEST-TLDS] .

2.3. Syntax Notation

This document uses the Augmented Backus Naur Form described in [ABNF]. A discussion of this is outside the bounds of this document, but it is expected that implementors will be able to quickly understand it with reference to the defining document.

This document is intended to be self-contained; all syntax rules used in it are defined within it, and a rule with the same name as one found in [MAIL] does not have the same definition. The lexical layer of [MAIL] is NOT, repeat NOT, used in this document, and its presence must not be assumed; notably, this document spells out all places where white space is permitted/required and all places where constructs resembling [MAIL] comments can occur.

NOTE: News parsers historically have been much less permissive than [MAIL] parsers.

Text in newsgroup names, header parameters, etc. is case-sensitive unless stated otherwise.

NOTE: This is at variance with [MAIL], which is case-insensitive unless stated otherwise, but is consistent with news historical practice and existing news software. See the comments on backward compatibility in section 1.

2.4. Language

Various constant strings in this document, such as header names and month names, are derived from English words. Despite their derivation, these words do NOT change when the poster or reader employing them is interacting in a language other than English. Posting and reading agents MAY translate as appropriate in their interaction with the poster or reader, but the forms that actually appear in articles MUST be the English-derived ones defined in this document.

3. Relation To [MAIL] (RFC 822 etc.)

The primary intent of this document is to completely describe the news article format. News articles were once considered as a subset of [MAIL]'s message format augmented by some new headers; this is no longer the case. News and [MAIL] have diverged. It is the intention of this document that gateways between [MAIL] and news still be capable of performing this function automatically.

[MAIL] and news do follow some of the same standards, however. In particular, the MIME standards apply equally to news articles.

4. Basic Format

4.1 Overall Syntax

Much of the syntax of News Articles is based on the corresponding syntax defined by [MESSFOR], which is deemed to have been incorporated into this standard as required. However, there are some important differences arising from the fact that [MESSFOR] does not recognise anything other than US-ASCII characters, that it does not recognise the MIME headers [RFC2045], and that it includes much syntax described as "obsolete".

The following syntactic forms supersede the corresponding rules given in [MESSFOR] and [RFC2045]:

```
text            = %d1-9 /           ; all octets except
                  %d11-12 /         ; US-ASCII NUL, CR and LF
                  %d14-255
ctext           = NO-WS-CTL /       ; all of <text> except
                  %d33-39 /         ; SP, HTAB, "(", ")"
                  %d42-91 /         ; and "\"
                  %d93-255
qtext           = NO-WS-CTL /       ; all of <text> except
                  %d33 /            ; SP, HTAB, "\" and <">
                  %d35-91 /
                  %d93-255
ftext           = %d33-57 /         ; all octets except
                  %d59-126 /        ; CTL, SP and ":"
                  %d128-255
token           = 1*<any ftext except tspecials>
tspecials       = "(" / ")" / "<" / ">" / "@"
                  "," / ";" / ":" / "
                  "/" / "[" / "]" / "?" / "="
```

Wherever in this standard the syntax is stated to be taken from [MESSFOR], it is to be understood as the syntax defined by [MESSFOR] after making the above changes, but NOT including any syntax defined in section 4 ("Obsolete syntax") of [MESSFOR]. Software compliant with this standard MUST NOT generate any of the syntactic forms defined in that Obsolete Syntax, although it MAY accept such syntactic forms. Certain syntax from the MIME specifications [RFC2045 et seq] is also considered a part of this Standard (see ...).

The following syntactic forms, taken from [RFC2234] or from [MESSFOR], are repeated here for convenience only:

```
ALPHA           = %x41-5A /         ; A-Z
                  %x61-7A           ; a-z
CR              = %x0D              ; carriage return
CRLF            = CR LF
DIGIT           = %x30-39           ; 0-9
HTAB            = %x09              ; horizontal tab
LF              = %x0A              ; line feed
SP              = %x20              ; space
NO-WS-CTL       = %d1-8 /           ; US-ASCII control characters
                  %d11 /            ; which do not include the
                  %d12 /            ; carriage return, line feed,
                  %d14-41 /         ; and whitespace characters
                  %d127
WSP             = SP / HTAB          ; Whitespace characters
FWS             = ([*WSP CRLF] 1*WSP)   ; Folding whitespace
comment         = "(" *([FWS] (ctext / quoted-pair / comment))
                  [FWS] ")"
CFWS            = *([FWS] comment) (([FWS] comment) / FWS )
<">             = %d34              ; quote mark
quoted-pair     = "\" text
quoted-string   = *CFWS <"> *(FWS (qtext / quoted-pair)) <"> *CFWS
unstructured    = *( [FWS] text )
```

4.2. Syntax of News Articles

The overall syntax of a news article is:

```
article         = 1*header separator body
header          = header-name ":" SP header-content CRLF
header-name     = 1*name-character *( "-" 1*name-character )
name-character  = ALPHA / DIGIT
header-content  = usenet-header-content / unstructured
usenet-header-content
     = <a header-content specifically defined in this standard>
separator       = CRLF
body            = *( *998text CRLF )
nonblank-text   = 1*( [FWS] nbtext )
nbtext          = qtext /           ; all of <text> except
                  "\" / <">         ; SP and HTAB
```

An article consists of some headers followed by a body. An empty line separates the two. The headers contain structured information about the article and its transmission. A header begins with a header-name identifying it, and can be continued onto subsequent lines as described in section 4.3.2. The body is largely unstructured text significant only to the poster and the readers.

NOTE: Terminology here follows the current custom in the news community, rather than the [MESSFOR] convention of referring to what is here called a "header" as a "header-field" or "field".

Note that the separator line must be truly empty, not just a line containing white space. Further empty lines following it are part of the body, as are empty lines at the end of the article.

4.3. Headers

4.3.1. Names and Contents

Despite the restrictions on header-name syntax imposed by the grammar, relayers and reading agents SHOULD tolerate header names containing any ASCII printable character other than colon (":", ASCII 58). [That brings it into line with <optional-field> as given in [MESSFOR].]

Header-names SHOULD be either those defined in this standard, or those defined in [MESSFOR], or those defined in any extension to either of these standards, or other names beginning with "X-". Software SHOULD NOT attempt to interpret headers not described in this standard or in its extensions. Relaying agents MUST pass them on unaltered and reading agents MUST enable them to be displayed, at least optionally.

Posters wishing to convey non-standard information in headers SHOULD use header-names beginning with "X-". No standard header name will ever be of this form. Reading agents SHOULD ignore "X-" headers, or at least treat them with great care.

The order of headers in an article is not significant. However, posting agents are encouraged to put mandatory headers (see section 5) first, followed by optional headers (see section 6), followed by "X-" headers and headers not defined in this standard or its extensions. Relaying agents MUST NOT change the order of the headers in an article.

Header-names are case-insensitive. There is a preferred case convention, which posters and posting agents SHOULD use: each hyphen-separated "word" has its initial letter (if any) in uppercase and the rest in lowercase, except that some abbreviations have all letters uppercase (e.g. "Message-ID" and "MIME-Version"). The forms used in this standard are the preferred forms for the headers described herein. Relaying and reading agents MUST, however, tolerate articles not obeying this convention.

4.3.2 Header Classes

There are four special classes of headers that may be present in an article: Experimental, Persistent, Comment, and Variant. All other headers are ephemeral. These classes are significant in how newsreaders and servers should treat them when encountered.

4.3.3 Experimental Headers

Experimental headers are headers which begin with "X-". They are to be used by newsreaders proposing new headers for some utility or for comments to be propogated with the article. There are no established headers that are considered experimental headers; an established header cannot be experimental.

Attempts to create new headers that are to be adopted as standard headers MUST begin their lives as experimental headers.

4.3.4 Persistent Headers

Persistent headers are headers which begin with "P-" (or "X-P-", hereafter referred to simply as "P- headers") which persist across followups either identically or by simple modification. Headers with this behavior include:

Newsgroups
Content is carried over into all followups. Modified by content of Followup-To header.

Subject
Content is carried over into all followups. Modified by prefixing with "Re: " if not already present. Also modified by user, often with a "(was:)" phrase preserving the previous content.

References
Content is carried over into all followups. Modified by appending content of Message-ID header.

NOTE: Though traditionally old newsreaders would treat Keywords as a persistent header, it is not a persistent header. More modern newsreaders do not treat it as such.

4.3.5. Variant Headers

Variant Headers are headers that are modified on articles when they are propogated. Variant headers have a "V-" prefix. Variant headers may be experimental ("X-V-"), persistent ("P-V-"), or both ("X-P-V-").

4.3.6. Header Classes

There are four special classes of headers that may be present in an article: Experimental, Persistent, Comment, and Variant. All other headers are ephemeral. These classes are significant in how newsreaders and servers should treat them when encountered.

4.3.6.1 Experimental Headers

Experimental headers are headers which begin with "X-". They are to be used by newsreaders proposing new headers for some utility or for comments to be propogated with the article. There are no established headers that are considered experimental headers; an established header cannot be experimental.

Attempts to create new headers that are to be adopted as standard headers MUST begin their lives as experimental headers.

4.3.6.2 Persistent Headers

Persistent headers are headers which begin with "P-" (or "X-P-", hereafter referred to simply as "P- headers") which persist across followups either identically or by simple modification. Headers with this behavior include:

Newsgroups

Content is carried over into all followups. Modified by content of Followup-To header.

Subject

Content is carried over into all followups. Modified by prefixing with "Re: " if not already present. Also modified by user, often with a "(was:)" phrase preserving the previous content.

References

Content is carried over into all followups. Modified by appending content of Message-ID header.

NOTE: Though traditionally old newsreaders would treat Keywords as a persistent header, it is not a persistent header. More modern newsreaders do not treat it as such.

4.3.6.3 Examples

```
Newsgroups: alt.test
Subject: Persistent Header Example
Message-ID: <001@news.site.example>
P-Author-IDs: <johnsmith-site.example-unique>
User-Agent: experimental/0.1g (P-Author-ID Compliant)

From: jane@site.invalid (Jane Smith)
Newsgroups: alt.test
Followup-To: misc.test
Subject: Re: Persistent Header Example
Message-ID: <002@news.site.example>
References: <001@news.site.example>
P-Author-IDs: <johnsmith-site.example-unique>
User-Agent: modern/1.2 (Author-ID non-Compliant; P- header compliant)
Keywords: persistance, good ideas

From: andrew@isp.invalid
```

```
Newsgroups: misc.test
Subject: Further example (was: Re: Persistent Header Example)
Message-ID: <001@news.isp.example>
References: <001@news.site.example> <002@news.site.example>
P-Author-IDs: <johnsmith-site.example-unique> <andrew@isp.example>
User-Agent: codeveloper/2.0b (Author-ID Compliant)
```

4.3.6.4 Comment Headers

Comment headers are headers that are strictly local and MUST NOT be propogated outside of a restricted subnet for local testing purposes. Comment headers have a prefix of "C-". Due to their limited scope, they MUST NOT be combined with any other prefix, such as "X-C-" headers. Headers with this behavior include:

Xref

Used by servers to keep track of crossposted articles' article numbers in the crossposted-to news groups in the local news spool as an aid to newsreaders marking such articles as read.

4.3.6.5. Variant Headers

Variant Headers are headers that are modified on articles when they are propogated. Variant headers have a "V-" prefix. Variant headers may be experimental ("X-V-"), persistent ("P-V-"), or both ("X-P-V-").

4.3.7. White Space and Continuations

[The following text is taken from [MESSFOR], adapted to the different terminology used for this standard.]

Each header is logically a single line of characters comprising the header-name, the colon with its following SP, and the header-content. For convenience, however, the header-content can be split into a multiple line representation; this is called "folding". The general rule is that wherever this standard allows for FWS (which includes CFWS, but not simply SP or HTAB) a CRLF followed by AT LEAST one SP or HTAB may instead be inserted. For example, the header:

```
Approved: modname@modsite.com(Acting Moderator of
comp.foo.bar)
```

can be represented as:

```
Approved: modname@modsite.com
     (Acting Moderator of comp.foo.bar)
```

NOTE: Though header-contents are defined in such a way that folding can take place between many of the lexical tokens, folding SHOULD be limited to placing the CRLF at higher-level syntactic breaks. For instance, if a header-content is defined as comma-separated values, it is recommended that folding occur after the comma separating the structured items, even if it is allowed elsewhere.

Folding MUST NOT be carried out in such a way that any line of a header is made up entirely of WSP characters and nothing else. [That is taken from a rather unsatisfactory line in section 3.2.4 of [MESSFOR] (which seems to allow WSP-only lines to arise from FWS but not from CFWS). The situation could arise where two FWS or CFWS could be adjacent, according to the syntax (I believe this is possible in [MESSFOR], which goes to show how sloppy their syntax is), or where FWS or CFWS is allowed at the end of a line.]

The colon following the header name on the start-line MUST be followed by white space, even if the header is empty. If the header is not empty, at least some of the content MUST appear on the start-line. Posting agents MUST enforce these restrictions, but relaying agents SHOULD accept even articles that violate them.

Posters and posting agents SHOULD use SP, not HTAB, where white space is desired in headers (some existing software expects this), and MUST use SP immediately following the colon after a header-name (this was an RFC 1036 requirement). Relaying agents SHOULD accept HTAB in all such cases, however.

Since the white space beginning a continuation line remains a part of the logical line, headers can be "broken" into multiple lines only at FWS or CFWS. Posting agents SHOULD not break headers unnecessarily (but see section 4.6).

4.3.8 Comments

Strings of characters which are treated as comments may be included in header contents wherever the syntactic element CFWS occurs. They consist of characters enclosed in parentheses. Such strings are considered comments so long as they do not appear within a quoted-string. Comments may be nested.

A comment is normally used to provide some human readable informational text, except at the end of an <address> which contains no <phrase>, as in

```
     fred@foo.bar.com (Fred Bloggs)
```

as opposed to

```
     "Fred Bloggs" <fred@foo.bar.com>
```

The former is a deprecated, but commonly encountered, usage and reading agents SHOULD take special note of such comments as indicating the name of the person whose <address> it is. In all other situations a comment is semantically interpreted as a single SP. Since a comment is allowed to contain FWS, folding is permitted within it as well as immediately preceding and immediately following it. Also note that, since quoted-pair is allowed in a comment, the parenthesis and backslash characters may appear in a comment so long as they appear as a quoted-pair. Semantically, the enclosing parentheses are not part of the comment token; the token is what is contained between the two parentheses.

Since comments have not hitherto been permitted in news articles, except in a few specified places, posters and posting-agents SHOULD NOT insert them except in those places. However, compliant software MUST accept them in all places where they are syntactically allowed.

4.3.9. Undesirable Headers

A header whose content is empty is said to be an empty header. Relaying and reading agents SHOULD NOT consider presence or absence of an empty header to alter the semantics of an article (although syntactic rules, such as requirements that certain header names appear at most once in an article, MUST still be satisfied). Posting and injecting agents SHOULD delete empty headers from articles before posting them; relaying agents MUST pass them untouched.

Headers that merely state defaults explicitly (e.g., a Followup-To header with the same content as the Newsgroups header, or a MIME Content-Type header with contents "text/plain; charset=us-ascii") or state information that reading agents can typically determine easily themselves (e.g. the length of the body in octets) are redundant and posters and posting agents SHOULD NOT include them.

4.4. Body

4.4.1. Body Format Issues

The body of an article MAY be empty, although posting agents SHOULD consider this an error condition (meriting returning the article to the poster for revision). A posting or injecting agent which does not reject such an article SHOULD issue a warning message to the poster and supply a non-empty body. Note that the separator line MUST be present even if the body is empty.

NOTE: Some existing news software is known to react badly to body-less articles, hence the request for posting and injecting agents to insert a body in such cases. The sentence "This article was probably generated by a buggy news reader" has traditionally been used is this situation.

Note that an article body is a sequence of lines terminated by CRLFs, not arbitrary binary data, and in particular it MUST end with a CRLF. However, relaying agents SHOULD treat the body of an article as an uninterpreted sequence of octets (except as mandated by changes of CRLF representation and by control-message processing) and SHOULD avoid imposing constraints on it. See also section 4.6.

4.4.2. Body Conventions

A body is by default an uninterpreted sequence of octets for most of the purposes of this standard. However, a MIME Content-Type header may impose some structure or intended interpretation upon it, and may also specify the character set in accordance with which the octets are to be interpreted.

NOTE: The syntax does not permit the NUL octet to appear in a body, and the octets CR and LF MUST ONLY occur together as CRLF. See also section 4.6 for limits on the length of a line.

It is a common practice for followup agents to enable the incorporation of the followed-up article (the "precursor") as a quotation. This SHOULD be done by prefacing each line of the quoted text (even if it is empty) with the character ">" (or preferably with "> "). This will result in multiple levels of ">" when quoted content itself contains quoted content. The followup agent SHOULD also precede the quoted content by an "attribution line" incorporating at least the name of the precursor's poster.

The following convention for attribution lines, whilst not mandated by this Standard, is intended to facilitate their automatic recognition and processing by sophisticated reading agents. The following fields describing the precursor should, if present, be in the given order.

A single Newsgroup name (the one from which the followup is being made) enclosed within <...> or <news:...>

The precursor's Message-ID enclosed within <...> or <news:...>

The precursor's poster's Name enclosed within "..."

The precursor's poster's Email address enclosed within <...> or <mailto:...>

The fields may be separated by arbitrary text, they may be folded in the same way as headers, and they should be terminated by a ":" followed by two CRLFs. Example:

```
On <comp.foo> in <12345678@foo.com> on 24 Dec 1997 16:40:20 +0000
"Joe D. Bloggs" <jdbloggs@foo.bar> wrote:
```

NOTE: The use of the standard character ">" facilitates automatic analysis of articles. The inclusion of the Message-ID in the attribution would enable reading agents to retrieve the precursor by clicking on it. However, readers are warned not to assume that attributions are accurate, especially within multiply nested quotations.

NOTE: Posters SHOULD edit quoted context to trim it down to the minimum necessary. However, followup agents SHOULD NOT attempt to enforce this beyond issuing a warning (past attempts to do so have been found to be notably counter-productive).

A "personal signature" is a short closing text automatically added to the end of articles by posting agents, identifying the poster and giving his network addresses, etc. If a poster or posting agent does append such a signature to an article, it MUST be preceded with a delimiter line containing (only) two hyphens (ASCII 45) followed by one SP (ASCII 32). The signature is considered to extend from the last occurrence of that delimiter up to the end of the article (or up to the end of the part in the case of a multipart MIME body). Followup agents, when incorporating quoted text from a precursor, SHOULD NOT include the signature in the quotation. Posting agents SHOULD discourage (at least with a warning) signatures of excessive length (4 lines is a commonly accepted limit).

4.5. Characters And Character Sets

Transmission paths for news articles MUST treat news articles as uninterpreted sequences of octets, excluding the values 0 (ASCII NUL) and 13 and 10 (ASCII CR and LF, which MUST only appear in the combination <CRLF> which denotes a line separator).

NOTE: this correspponds to the range of octets permitted for MIME "8bit data" [RFC-2045].

An octet, or a sequence of octets, may represent a character in some Coded Character Set (CCS) [RFC-2130] as determined by some Character Encoding Scheme (CES) [RFC-2130].

If it comes to a relaying agent's attention that it is being asked to pass an article using the Content-Transfer-Encoding "8bit" to a relaying agent that does not support it, it SHOULD report this error to its administrator. It MUST refuse to pass the article and MUST NOT re-encode it with different MIME encodings.

NOTE: This strategy will do little harm. The target relaying agent is unlikely to be able to make use of the article on its own servers, and the usual flooding algorithm will likely find some alternative route to get the article to destinations where it is needed.

4.5.1. Character Sets within Article Headers

Within article headers, the CES is UTF-8 [ISO-10646 or RFC-2279] and hence the CCS is the Universal Multiple-Octet Coded Character Set (UCS) [ISO-10646] (which is essentially a superset of Unicode [UNICODE] and expected to remain so). However, interpreting the octets directly as ASCII characters should ensure correct behaviour in most situations.

NOTE: UTF-8 is an encoding for 16bit (and even 32bit) character sets with the property that any octet less than 128 immediately represents the corresponding ASCII character, thus ensuring upwards compatibility with previous practice. Non-ASCII characters from UCS are represented by sequences of octets greater than 127. Only those octet sequences explicitly permitted by [RFC 2079] shall be used. UCS includes all characters from the ISO-8859 series of characters sets [ISO-8859] (which includes all Greek and Arabic characters) as well as the more elaborate characters used in Japan and China. See the following section for the appropriate treatment of UCS characters by reading agents.

Notwithstanding the great flexibility permitted by UTF-8, there is need for restraint in its use in order that the essential components of headers may be discerned using reading agents that cannot present the full UCS range. In particular, header-names MUST be in ASCII, and certain other components of headers, as defined elsewhere in this standard - notably <identifier>s (as in <message-id>s), <date-time>s, <domain>s <addr-spec>s and <path-item>s - MUST be in ASCII. <Comment>s, <phrase>s (as in <address>es) and <unstructured>s (as in <subject>s) MAY use other character sets. For <newsgroup-name>s see below.

Where the use of non-ASCII characters, encoded in UTF-8, is permitted as above, they MAY also be encoded using the MIME mechanism defined in RFC-2047 [RFC-2047], but this usage is deprecated within news articles (even though it is required in mail messages) since it is less legible in older reading agents which support neither it nor UTF-8. Nevertheless, reading agents SHOULD support this usage, but only in those contexts explicitly mentioned in [RFC-2047].

4.5.2 Character Sets within Article Bodies

Within article bodies, the CES and CCS implied by any Content-Transfer-Encoding and Content-Type headers [RFC-2045] SHOULD be applied by reading agents. In the absence of such headers, reading agents cannot be relied upon to display correctly more than the ASCII characters. [Observe that reading agents are not forbidden to "guess", or to interpret as UTF-8 regardless, which would be the simplest course for them to take.]

NOTE: It is not expected that reading agents will necessarily be able to present characters in all possible character sets, although they MUST be able to present all ASCII characters. For example, a reading agent might be able to present only the ISO-8859-1 (Latin 1) characters [ISO-8859], in which case it SHOULD present undisplayable characters using some distinctive glyph, or by exhibiting a suitable warning. Older reading agents that do not understand MIME headers or UTF-8 should be able to display bodies in ASCII (with some loss of human comprehensibility) except possibly when the Content-Transfer-Encoding is "8bit".

NOTE: Be warned that it will never be safe to send raw binary data in the body of news articles, because the presence of ASCII NUL and changes of <CRLF> representation will inevitably corrupt it. Such data MUST be encoded (e.g. by using Content-Transfer-Encoding: base64).

Posters SHOULD avoid using control characters in ASCII (or other CCSs) except for tab (ASCII 9), formfeed (ASCII 12), and backspace (ASCII 8). Tab signifies sufficient horizontal white space to reach the next of a set of fixed positions; posters are warned that there is no standard set of positions, so tabs should be avoided if precise spacing is essential. Formfeed signifies a point at which a reading agent SHOULD pause and await reader interaction before displaying further text. Backspace SHOULD be used only for underlining, done by a sequence of underscores (ASCII 95) followed by an equal number of backspaces, signifying that the same number of text characters following are to be underlined. Posters are warned that underlining is not available on all output devices and is best not relied on for essential meaning. Reading agents SHOULD recognize underlining and translate it to the appropriate commands for devices that support it. Reading agents MUST NOT pass other control characters or escape sequences unaltered to the output device.

Followup agents MUST be careful to apply appropriate encodings to the outbound followup. A followup to an article containing non-ASCII material is very likely to contain non-ASCII material itself.

4.6. Size Limits

The syntax provides for the lines of a body to be up to 998 octets in length, not including the CRLF. All software compliant with this standard MUST support lines of at least that length, both in headers and in bodies, and all such software SHOULD support lines of arbitrary length. In particular, relaying agents MUST transmit lines of arbitrary length without truncation or any other modification.

NOTE: The limit of 998 octets is consistent with the corresponding limit in [MESSFOR].

In plain-text messages (those with no MIME headers, or those with a MIME Content-Type of text/plain) posting agents SHOULD encourage the practice of keeping the length of body lines to within 79 characters at most, and preferably to within 72 characters (to allow room for quoting in followups). However, posting agents MUST permit the poster to include longer lines if he so insists.

NOTE: Plain-text messages are intended to be displayed "as-is" without any special action (such as automatic line splitting) on the part of the recipient. The limit (72 or 79) is expressed as a number of characters (as they will be displayed by a reading agent) rather than as the number of octets used to encode them.

Posting agents SHOULD fold headers by inserting CRLF followed by 1*WSP at positions (preferably higher-level ones - see 4.3.2) where this is syntactically allowed so as to keep, so far as is possible, all header lines within 79 characters. Likewise, injecting agents SHOULD fold any headers generated automatically by themselves. Relaying agents MUST NOT fold header lines (i.e. they must pass on the folding as received).

NOTE: There is NO restriction on the number of lines into which a header may be split, and hence there is NO restriction on the total length of a header (in particular it may, by suitable folding, be made to exceed the 998 octets restriction pertaining to a single header line).

NOTE: This standard provides no upper bound on the overall size of a single article, but neither does it forbid relaying agents from dropping articles of excessive length. It is, however, suggested that any limits thought appropriate by particular agents would be more appropriately expressed in megabytes than in kilobytes.

4.7. Example

Here is a sample article:

```
Path: server.example,unknown.site2.example@site2.example,
  relay.site.example,site.example,injector.site.example%jsmith
Newsgroups: example.announce,example.chat
Message-ID: <9urrt98y53@site.example>
From: Ann Example <a.example@site1.invalid>
Subject: Announcing a new sample article.
Date: Fri, 27 Mar 1998 12:12:50 +1300
Approved: example.announce moderator <jsmith@site.invalid>
Followup-To: example.chat
Reply-To: Ann Example <a.example+replies@site1.example>
Expires: Wed, 22 Apr 1998 12:12:50 -0700
Organization: Site1, The Number one site for examples.
User-Agent: ExampleNews/3.14 (Unix)
Keywords: example, announcement, standards, RFC 1036, Usefor
Summary: The URL for the next standard.


Just a quick announcemnt that a new standard example article has been
released; it is in the new USEFOR draft obtainable from ftp.ietf.org.

Ann.

--
Ann Example <a.example@site1.invalid>       Sample Poster to the Stars
"The opinions in this article are bloody good ones" - from J Clarke.
```

5. Mandatory Headers

An article MUST have one, and only one, of each of the following headers: Date, From, Message-ID, Subject, Newsgroups, Path.

NOTE: [MAIL] specifies (if read most carefully) that there must be exactly one Date header and exactly one From header, but otherwise does not restrict multiple appearances of headers. (Notably, it permits multiple Message-ID headers!) This appears singularly useless, or even harmful, in the context of news, and much current news software will not tolerate multiple appearances of mandatory headers.

Note also that there are situations, discussed in the relevant parts of section 6, where References, Sender, or Approved headers are mandatory. In control articles, specific values are required for certain headers.

In the discussions of the individual headers, the content of each is specified using the syntax notation. The convention used is that the content of, for example, the Subject header is defined as <Subject-content>.

NOTE: see also Section 7.1.1

5.1. Date

The Date header contains the date and time that the article was submitted for transmission. The content syntax is defined in the Message Format Standard [MESSFOR].

```
Date-content = date-time
```

5.2. From

The From header contains the electronic address(es), and possibly the full name, of the article's author(s) . The format of the From header is defined in the Message Format Standard [MESSFOR].

All mailboxes in the From-content field MUST either belong to the posters(s) of the article (or the poster(s) are authorized by the owners to use the mailboxes) or end in the top level domain of ".invalid".

```
From-content = mailbox-list
```

5.2.1 Examples:

```
From: John Smith <jsmith@site.example>
From: John Smith <jsmith@site.example>, dave@isp.example
From: John Smith <jsmith@site.example>, andrew@isp.example,
   fred@site2.example
From: Jan Jones <jan@please_setup_your_software_correctly.invalid>
From: Jan Jones <joe@anonymous.invalid>
From: dave@isp.example (Dave Smith)
```

NOTE: the last example is in an obsolete syntax.

5.3. Message-ID

The Message-ID header contains the article's message ID, a unique identifier distinguishing the article from every other article. The format of the Message-ID header is defined in the Message Format Standard [MESSFOR] . An article's message ID MUST be unique and MUST NEVER be reused.

```
     Message-ID-content = msg-id
```

5.4. Subject

The Subject field contains a short string identifying the topic of the message. When used in a followup, the field body SHOULD start with the string "Re: " (a "back reference") followed by the contents of the pure-subject of the precursor.

```
subject-content = [ back-reference ] pure-subject CRLF
pure-subject = nonblank-text
back-reference = %x52.65.3A.20        ; which is a case-sensitive
                                       "Re: "
```

The pure-subject MUST NOT begin with "Re: ". The default subject-content of a followup is the string "Re: " followed by the contents of the pure-subject of the precursor. Any leading "Re: " in the pure-subject MUST be stripped.

Followup agents SHOULD remove instances of non-standard back-reference (such as "Re(2): ", "Re:", "RE: ", or "Sv: ") from the subject-content when composing the subject of a followup and add a correct back-reference in front of the result.

Followup agents MUST NOT use any other string except "Re: " as a back reference. Specifically, a translation of "Re: " into a local language or usage MUST NOT be used.

Agents SHOULD NOT depend on nor enforce the use of back references by followup agents. For compatibility with legacy news software the subject-content of a control message MAY start with the string "cmsg ", non-control messages MUST NOT start with the string "cmsg ".

5.4.1 Examples:

In the following examples, please note that only "Re: " is mandated by this DRAFT. "was: " is a convention used by many English-speaking posters to signal a change in subject matter. Software should be able to deduce this information from References.

```
Subject: Film at 11.
Subject: Re: Film at 11
Subject: Use of Godwin's law considered harmful (was: Film at 11)
Subject: Godwin's law (was: Film at 11)
```

```
Subject: Re: Godwin's law (was: Film at 11)
```

5.5. Newsgroups

The Newsgroups header's content specifies which newsgroup(s) the article is posted to:

```
Newsgroups-content = newsgroup-name *( ng-delim newsgroup-name)
newsgroup-name = *FWS component *( "." component ) *FWS
component = component-start [*component-rest component-start]
component-start = lowercase / digit
lowercase = <Unicode Letter, Lowercase> / <Unicode Letter, Other>
uppercase = <Unicode Letter, Uppercase> / <Unicode Letter, Titlecase>
digit = <Unicode Number, Decimal Digit> / <Unicode Number, Other>
component-rest = component-start / "+" / "-" / "_"
ng-delim = ","
```

where the <Unicode ...> items are as described in [UNICODE].

The inclusion of folding white space within a newsgroup-name is a newly introduced feature in this standard. It MUST be accepted by all conforming implementations (relaying agents, serving agents and reading agents). On the other hand, posting agents MUST NOT generate such whitespace and injecting agents MUST NOT accept such whitespace (except for experimental postings to 'test' newsgroups or within cooperating subnets) until after AGREED IMPLEMENTATION DATE. After AGREED IMPLEMENTATION DATE such agents MAY generate such whitespace anywhere and SHOULD generate it in the form of <CRLF WS> so as to keep the length of lines in the relevant headers (notably Newsgroups and Followup-To) to no more than than 79 characters. Before AGREED IMPLEMENTATION DATE, injecting agents MAY reformat such headers by removing whitespace inserted by the posting agent, but relaying agents MUST NOT do so.

A newsgroup name consists of one or more components. Components MAY contain non-ASCII letters, but these MUST be encoded in UTF-8 and not according to RFC-2047. A component MUST contain at least one letter (and must, according to the syntax, begin and end with a letter or digit). Components SHOULD begin with a letter. Composite characters (made by overlaying one character with another) and format characters, as allowed in certain parts of Unicode and needed by certain languages, must use whatever canonical conventions apply to those parts of Unicode (such conventions are not defined in this Standard). The use of "_" in a component is deprecated. Serving agents MAY refuse to accept newsgroups using that component.

NOTE: Components composed entirely of digits would cause problems for the commonly used implementation technique of using the component as the name of a directory, whilst also using sequential numbers to distinguish the articles within a group.

NOTE: Uppercase letters MUST NOT be used. Although converting ASCII uppercase letters to their lowercase counterparts is straightforward enough, it would be unreasonable to expect software to do the same in parts of Unicode for which it was not configured (in general, a table lookup would be required). Thus software MAY attempt to convert uppercase letters according to the mappings defined by [UNICODE], but this behaviour is not required.

Whilst there is no longer any technical reason to limit the length of a component (formerly, it was limited to 14 characters) nor to limit the total length of a newsgroup-name, it should be noted that these names are also used in the newsgroups line (...) where an overall limit applies, and moreover excessively long names can be exceedingly inconvenient in practical use. Those responsible for the management of the various netnews hierarchies SHOULD therefore set reasonable limits for the length of a component and of a newsgroup name. In the absence of such explicit policies, figures of 30 characters and 72 characters respectively are recommended.

NOTE: The newsgroup-name as encoded in UTF-8 should be regarded as the canonical form. Reading agents may convert it to whatever character set they are able to display (see 4.5.2) and serving agents may possibly need to convert it to some form more suitable as a filename. Simple algorithms for both kinds of conversion are readily available.

Posters SHOULD use only the names of existing newsgroups in the Newsgroups header, because newsgroups are not created simply by being posted to. However, it is legitimate to cross-post to newsgroup(s) which do not exist on the posting agent's host, provided that at least one of the newsgroups DOES exist there, and followup agents MUST accept this (posting agents MAY accept it, but SHOULD at least alert the poster to the situation and request confirmation). Relaying agents MUST NOT rewrite Newsgroups headers in any way, even if some or all of the newsgroups do not exist on the relaying agent's host.

5.5.1 Forbidden newsgroup names

The following newsgroup-names MUST NOT be used:

Newsgroup-names having only one component (reserved for newsgroups whose propagation is restricted to a single host, or the administrative equivalent).

"poster" (because it has special meaning in the Followup-To header (see section 6.1).)

"newsgroups" (likewise)

"junk" (frequently used for pseudo-newsgroups internal to serving agents)

"control" (likewise)

Any newsgroup-name beginning with "control." (likewise)

Any newsgroup-name containing the component "ctl" (likewise)

"to" or any newsgroup-name beginning with "to." (reserved for test messages sent on an essentially point-to-point basis (see also the ihave/sendme protocol described in section 7.2)

Any newsgroup-name containing the component "all" (because this is used as a wildcard in some implementations)

A newsgroup SHOULD NOT appear more than once in the Newsgroups header. The order of newsgroup names in the Newsgroups header is not significant.

5.6 Path

The Path header shows the route a message took from its entry into the USENET system to the current system. It is a list of site identifiers with the origin on the right. Each relaying, injecting or serving agent that processes the article adds one or more entries to this header. Aside from tracing the route articles take in moving over the network, Path is used primarily to allow relaying systems to not send articles to sites known to already have them, in particular the site they came from. This improves the efficiency of links. Path is also used for USENET statistics gathering and flow tracking. Finally the presence of a "%" delimiter in the Path header can be used to identify an article injected in conformance with this standard.

5.6.1 Format

```
path-content    =       old-path / new-path

old-id          =       1*( ALPHA / digit / "-" | "." | "_")
```

```
old-path        =       old-id *(punctuation old-id)
punctuation     =       LWSP / %x21-2f / %x3a-40 / %x5b-60 / %x7b-7f
                              ; These are ! " # $ % & ' ( ) *
                              ;   + , - . / : ; < = > ? @ [ \
                              ;   ] ^ _ ` { | } ~ DEL
new-delims      =       [FWS] ("@" / "/" / "," ) [FWS]
new-path        =       post-injection "%" pre-injection
delim-plus-id   =       [FWS] "!" [FWS] old-id
                        / new-delims site-id
post-injection  =       *(site-id 1*new-delims) site-id
pre-injection   =       site-id *delim-plus-id
site-id         =       ALPHA word      ; UUCP name
                        / ALPHA         ; for "x" tail entry
                        / "." word      ; other registered name
                        / <FQDN>        ; as per RFC 1034
                        / <dotted-quad> ; numeric IP address rep
                                        ; specified in rfc820 etc.
                        / "[" dotted-quad "]"
                        / "[" <ipv6-numeric> "]" ; per RFC1884
word            =       1*(ALPHA / digit / "-" / "_")
```

5.6.2 **Adding an entry to the Path header.**

When a system receives a message from another system, it MUST add its own unique name (path-identity or site-id) and a delimiter to the beginning of the Path string. In addition, if needed, folding-whitespace MAY be added.

The path-identity added MUST be unique. To this end it should be one of:

1. A name registered previously in the UUCP maps database (found in the newsgroup comp.mail.maps), containing no dot character.

3. The fully qualified domain name or MX record, retrievable via the Internet DNS service.

4. An encoding of an IP address -- dotted quad or for IPv6 as per RFC1884. These encodings using SHOULD NOT be used prior to draft-implementation-date.

Whichever form is chosen, a site SHOULD use a form which can be verified using one of the schemes described below by all sites to which it will forward news articles. If all forwarding is by NNTP or other internet based protocols, then the FQDN or IP address encodings are advised. For the purposes of comparison, FQDN entries should be put in an all-lower-case canonical form.

Because RFC1036 specified any punctuation or whitespace could act as delimiter, programs SHOULD accept this, with the exception that IPv6 addresses containing colons MUST be treated as a single unit. Modern programs MUST generate only the set "!,%@" plus optional additional whitespace.

When a site receives an article from another site, it SHOULD (MUST after draft-implementation-date), verify the identity of the source site. When processing an article from a source, the leftmost entry of the Path line should be extracted, converted to a canonical form, and tested to see if it matches the canonical form of the verified identity of the source. If it does, a "," should be used as the delimiter, and thus the comma, and then the receiving site's path-identity MUST be prepended to the Path line.

The method of verification is up to the site. Any method of suitable authenticity may be chosen, with the consideration that in the event of problems at the source site, the relaying site may be called upon to reliably identify it.

If the leftmost entry does not match the verified identity of the source, then the receiving site should prepend an "@" delimiter, then a simple form of the verified identity of the source, then a "," delimiter and then the receiving site's own path-identity. This adding of two identities to the line should not be done if the provided and verified identities match. For articles received from an internet source, the unique 32 bit IPv4 address or properly verified FQDN, whichever is shorter, is encouraged for the generated ID.

5.6.3 The tail Entry

For historical reasons, the rightmost entry in the Path string generated by most systems is not a site name, but a "user name". However, the Path string is not an E-mail address and MUST NOT be used to contact the user. Injecting agents MAY place any string here that is not a path-identity. If no meaning is anticipated the string "x" SHOULD be used.

RFC1036 suggested that the last entry could be a site name, requiring software to check it when feeding, but said it also should have a user-id for very old systems. As of this specification, a systems MUST NOT treat the tail entry as a path-identity.

Typically this field will be the only entry on the Path string generated by a poster, or if not generated by the posting-agent, by the injecting agent, which will prepend a "%" and then its own verifiable path-identity. The percent divides the verified part of the Path line from any entries provided prior to injection into the news network. There may be more than one entry to the left of the percent, and all but the last are to be treated as sites.

Injecting Agents SHOULD use the tail entry for local authentication information on the source of an article. For example, if they wish to store an encoding of the IP address of a source machine connecting to do the injection, and/or the UID of an invoking user or any other such information, they may encode it in the tail entry, provided they do so in a manner that will not match any site identifier. (e.g. ending with a dot) .

5.6.4 The Injecting Agent Entry

The injecting agent's path identity is a special case. This identity MUST be a FQDN which can be used as a domain for E-mail connections (ie. it should have either an A or MX record). See the Duties of an Injection Agents section 7.1 and RFC 2142.

5.6.5 Delimiter Summary

A summary of delimiters and the meaning they imply for the name on the right, or in addition, the name to the left.

, Verified or generated identity.

@ Name failed verification test. Name on left is identity generated by site further to the left.

% Optional pre-injection entries followed by tail entry. Commonly just the tail entry, either "x" or an encoding of login identity. Name on left is FQDN of site that handles mail for Injecting Agent. The presence of two "%" in a path indicates a double-injected error.

! Entry is unverified. Identity on left is an old-style system not conformant with this specification.

Folding Whitespace MUST NOT be used as the sole delimiter.

Other Treat as "!" as per RFC1036

"/" Reserved for future use, treat as ","

; Semicolon is reserved for the generation of extensible headers.

: The colon is a valid delimiter for legacy systems, however, inside an IPv6 numeric address, surrounded in square brackets, it is a part of the path-identifier.

_ This should not be treated as punctuation (a delimiter), contrary to RFC1036. Treat as part of identifiers.

5.6.6 Other formatting Issues

The Path header MUST NOT be truncated.

Whitespace MAY be present in the Path to make it easier to represent. However, there is no requirement to do so. Whitespace MUST not be used as a delimiter.

5.6.6.1 Use of "!"

Old USENET relaying and injecting programs almost all delimit Path: entries with the "!" delimiter, and these entries are not verified. As such, the presence of "%" as a delimiter will indicate the article was injected by software conforming to this standard, and the presence of "!" as a delimiter will indicate the message passed through systems developed prior to this standard. Prior to the draft-implementation-date, messages with mixed sets of delimiters will be common. After that date, all messages should have no "!" delimiters prior to the "%" delimiter.

5.6.7 Suggested Verification Methods

Sites attempting to verify an incoming entry SHOULD take the following approaches for common transports. They are not required, but not following them may lead to wasteful double-entry Path additions.

If the incoming article arrives through some protocol local to the site, such as UUCP, that protocol MUST include a means of verifying the article source site, and this should match. In UUCP implementations, commonly each incoming connection has a unique login name and password; that login name could be used to build a suitable verified identifier.

Here is an example of a suitable verification method for an article arriving via a TCP/IP protocol such as via NNTP:

1. If it is an encoding of an IP address, it should be decoded into a canonical form. If that address does not match the source's IP, a reverse-DNS (in-addr.arpa PTR record) lookup should be done on the provided address, followed by a regular DNS "A" record lookup on the returned name. That A record may contain several IP addresses. So long as one matches the IP address from the path, and another matches the source IP address, this is considered a match.

2. If it is a internet DNS style FQDN, then the name should be looked up with DNS. The A records MUST contain an IP address that is the verified address of the source.

3. (It should be noted that when generating a name after a non-match, if an FQDN is desired, simply doing a reverse DNS (PTR) lookup on the IP address is not sufficient to generate the FQDN. The returned name must be mapped back to A records to assure it matches the source's IP address.)

5.6.8 Issues

There is no firm way to tell a path entry generated by new software, and one generated by old software assuming that any delimiter is valid. However, use of "!" by old software has become effectively universal.

Sites are not strictly required to use a standard form for their path entry, but if they don't, path lines out of that site get longer due to the adding of the identity. However, groups of associated sites wanting a common identity may decide to use that and let the receiver add the specific site.

6. Optional Headers

The headers appearing in this section have established meanings. They MUST be interpreted according to the definitions made in this document. None of them are required to appear in every article. All of the headers appearing in this document MUST NOT appear more than once in an article. Headers not appearing in this document (i.e. X-headers, headers defined by cooperating subnets) are exempt from this requirement. See "Responsibilities of Agents" for a clear picture.

6.1 Followup-To

The Followup-To header contents specify which newsgroup(s) followups should be posted to:

```
Followup-To-content = Newsgroups-content / "poster"
```

The syntax is the same as that of the Newsgroups content, with the exception that the magic word "poster" means that followups should be mailed to the article's reply address rather than posted. In the absence of Followup-To, the default newsgroup(s) for a followup are those in the Newsgroups header and for this reason the Followup-To header should not be included if it just duplicates the Newsgroups header.

6.2 Expires

The Expires header content specifies a date and time when the article is deemed to be no longer useful and should be removed ("expired"). The content syntax is the same as that of the Date content which is defined in the Message Format Standard [MESSFOR] .

```
     expires-content = date-time
```

A Expires header SHOULD only be used in an article if the requested expiry time is earlier or later than the default would normally be for that article. Local policy for each serving agent will dictate when this header is obeyed and authors SHOULD NOT depend on it being completely followed.

6.3. Reply-To

The Reply-To header content specifies a reply address(es) to be used for personal replies for the author(s) of the article when this is different from the author's address(es) given in the From header. The format of the Reply-To header is defined in the Message Format Standard [MESSFOR] .

In the absence of Reply-To, the reply address(es) is the address(es) in the From header. For this reason a Reply-To SHOULD NOT be included if it just duplicates the From header.

Use of a Reply-To header is preferable to including a similar request in the article body, because reply agents can take account of Reply-To automatically.

"Reply-To: <> " MAY be used to indicate that the poster does not wish to recieve email replies.

```
      Reply-To-content = From-content
```

6.3.1 Examples:

```
Reply-To: John Smith <jsmith@site.example>
Reply-To: John Smith <jsmith@site.example>, dave@isp.example
Reply-To: John Smith <jsmith@site.example>, andrew@isp.example,
   fred@site2.example
Reply-To: Please not not reply <>
```

6.4. References

The References header content lists optionally CFWS-separated message ids of precursors. The format of the References header is defined in the Message Format Standard [MESSFOR].

A followup MUST have a References header, and an article that is not a followup MUST NOT have a References header. In a followup, if the precursor did not have a References header, the followup's References content MUST be formed by the message ID of the precursor. A followup to an article which had a References header MUST have a References header containing the precursor's References content, plus the precursor's message ID appended to the end of the list (separated from it by optional CFWS).

Followup Agents SHOULD NOT trim message ids out of the References content unless the number of message ids exceeds 31 in which case message ids SHOULD be trimmed until there are only 31.

Trimming SHOULD be done by removing the sixth (6th) message-id and any incomplete or otherwise broken message-ids. If Followup Agents trim any message-ids out of the References content, then they MUST leave the first five and the last nine message ids and they SHOULD also leave any message ids mentioned in the body of the article intact.

NOTE: Software writers should be aware that the number of messages ids in this header may exceed 31 and software must be able to handle this without problem.

```
References-content = msg-id [msg-id...]
```

6.4.1 Examples:

```
References: <i4g587y@site1.example>
References: <i4g587y@site1.example> <kgb2231+ee@site2.example>
References: <i4g587y@site1.example><kgb2231+ee@site2.example>
   <222@site1.example><87tfbyv@site7.example><67jimf@site666.example>
References: <i4g587y@site1.example> <kgb2231+ee@site2.example>
   <tisjits@smeghead.example>
```

6.5. Control

The Control header content marks the article as a control message, and specifies the desired actions (other than the usual ones of filing and passing on the article):

Control-content = verb *(FWS argument) verb = 1*(ALPHA / DIGIT) argument = 1* ftext

The verb indicates what action should be taken, and the argument(s) (if any) supply details. In some cases, the body of the article may also contain details. The next section describes the standard verbs.

6.6. Control Messages

The following sections document the group control messages. "Message" is used herein as a synonym for "article" unless context indicates otherwise. Group control messages are a special class of control messages, that request the group configuration on a server be updated.

All of the group control messages MUST have an Approved header (section 6.10). They SHOULD use one of the authentication mechanisms defined in section TBD.

The execution of the actions requested by control messages is subject to local administrative restrictions, which MAY deny requests or refer them to an administrator for approval. The descriptions below are generally phrased in terms suggesting mandatory actions, but any or all of these MAY be subject to local administrative approval (either as a class or case-by-case). Analogously, where the description below specifies that a message or portion thereof is to be ignored, this action MAY include reporting it to an administrator.

Relaying Agents MUST propagate even control messages they do not understand.

In the following sections, each type of control message is defined syntactically by defining its arguments and its body. For example, "cancel" is defined by defining cancel-arguments and cancel-body.

6.6.1 The "newgroup" Control Message

```
newgroup-ctrl         = "newgroup" FWS groupname [ FWS flags ]
flags                 = "moderated"
groupname             ; defined in [NEWS]
```

The "newgroup" control message requests the specified group be created or changed. The text "moderated" is appended to mark the group as moderated. The message contains a "multipart/newsgroupinfo" (section 6.6.1 body) part containing machine- and human-readable information about the group.

The newgroup command is also used to update the description line or moderation status of a group.

NOTE: It is also possible to send newgroups for existing groups that don't change anything to ensure the group exist on all systems ("booster" newgroups). Implementations might want to test for this condition before attempting to update their configuration.

6.6.1.1 multipart/newsgroupinfo

The "multipart/newsgroupinfo" body structure contains information about a (new) newsgroup.

The MIME content type definition of "multipart/newsgroupinfo" is:

```
MIME type name:           multipart
MIME subtype name:        newsgroupinfo
Required parameters:      boundary (see [MIME2])
Optional parameters:      none
Encoding considerations: "7bit" or "8bit" is sufficient and
                          MUST be used to maintain compatibility.
Security considerations:  to be added
```

A "multipart/newsgroupinfo" body part contains the following subparts:

1. An "application/newsgroupinfo" part (section 6.6.1.2) containing the name and description line of the group(s). This part is mandatory.

2. Other parts containing useful information about the backgrounds of newsgroup message.

3. Parts containing initial named articles for the newsgroup. See section 6.6.1.3 for details.

6.6.1.2 application/newsgroupinfo

The "application/newsgroupinfo" body part contains a short information on a newsgroup, i.e. the group's name, it's description and the moderation flag.

NOTE: This part has a format that makes the whole "multipart/newsgroupinfo" structure compatible with [1036BIS].

The MIME content type definition of "application/newsgroupinfo" is:

```
          MIME type name:            application
          MIME subtype name:         newsgroupinfo
          Optional parameters:       none
          Encoding considerations:  "7bit" or "8bit" is sufficient and
                                     MUST be used to maintain compatibility.
                                     Note that the descriptions may use
[MIME3].
          Security considerations: to be added
```

The content of the "application/newsgroupinfo" body part is defined as:

```
 groupinfo-body     = descriptor-tag CRLF 1*( description-line CRLF )
 descriptor-tag     = %x46.6F.72 SP %x79.6F.75.72 SP
                      %x6E.65.77.73.67.72.6F.75.70.73 SP
                      %x66.69.6C.6E.3A
                    ; case sensitive "For your newsgroups file:"
description-line    = newsgroup-name [ 1*WSP description
                    [ 1*SP group-flags ] ]
description         = nonblank-text / encoded-word
moderation-flags    = [ moderated-literal ]
moderated-literal   = %x28.4D.6F.64.65.72.61.74.65.64.29
                       ; case sensitive "(Moderated)"

group-flags = [ "<" addr-spec ">" 1*SP ] "(Moderated)"
```

The "application/newsgroupinfo" is used in conjunction with the "newgroup" (section 6.6.1) and "mvgroup" control messages (section 6.6.3) as part of a "multipart/newsgroupinfo" (section 6.6.1) MIME structure.

Moderated newsgroups SHOULD be marked by appending the case sensitive text " (Moderated)" at the end.

NOTE: Due to the line lenght limit in [MAIL], [NEWS] and [NNTP], a description line can have a maximum length of 998 octets.

NOTE: In some hierarchies, there exist conventions that set a far lower limit, often in characters.

NOTE: Usually, the description length is limited in a way that the newsgroup name, the tab (interpreted as an 8-character tab that takes one at least to column 24) and the description without flags fit into the first 79 characters.

NOTE: Servers that use an "newsgroups" file will store the group descritpions there as is, i.e. without any conversion of charsets or encoding.

NOTE: The descriptions will also be used with the [NNTP] LIST NEWSGROUPS command. The descriptions will be sent as is, i.e. without any conversion of charsets or encoding.

6.6.1.3 Initial Named Articles

Some parts of a multipart/newsgroupinfo structure MAY contain an initial set of named articles. These parts are identified by the Article-Name header just like normal named article postings. The named articles are filed separately as single postings, where the headers of the enclosing control message are copied to every part that contains a named article except that:

Content-* and Article-* headers MUST be taken from the body part.

The message id MUST be changed by inserting /partX before the @ sign, where X is the number of the body part, starting with 0. The Control header of the enclosing message header MUST be stripped. It MAY be replaced by a "Control: named" header. Signatures (Auth, X-Auth...) of the enclosing message SHOULD be stripped. They MAY be replaced by a signature of the own site.

The resulting articles are for internal use of the server and its users only, they MUST NOT, repeat MUST NOT be forwarded to other sites.

Nested multipart/* structures are allowed, they are not recursively expanded to separate articles.

6.6.2 The "rmgroup" Control Message

```
rmgroup-ctrl  = "rmgroup" FWS groupname
```

The "rmgroup" control message requests the specified group be removed from the list of valid groups. The Content-Type of the body is unspecified; it MAY contain anything, usually an explaining text.

NOTE: It is also possible to send rmgroups for nonexisting, bogus groups to ensure the group is removed on all systems ("booster" rmgroups). Implementations might want to test for this before attempting to update their configuration.

6.6.3 The "mvgroup" Control Message

```
mvgroup-ctrl  = "mvgroup" FWS ( mvgrp-groups / mvgrp-hrchy)
mvgrp-groups  = groupname [ FWS groupname ]
mvgrp-hrchy   = groupnamepart ".*" FWS groupnamepart
groupnamepart = groupname    ; syntactically
```

6.6.3.1 Single group

The "mvgroup" control message requests the first specified group to be moved to the second group. The message contains a "multipart/newsgroupinfo" (section 6.6.1.2) body part containing machine- and human-readable information about the new group.

When this message is received, the new group SHOULD be created and all articles, including named articles, SHOULD be copied or moved to the new group, then the old, now empty group SHOULD be deleted.

NOTE: For servers that use a file system directory structure to organize message storage, this operation is quite efficiently implemented as a single directory rename operation.

If the old group does not exist, the message is ignored unless the new group does not exist either, in which case the new group is created just as for a "newgroup" message.

An indication that the old group was replaced by the new group MAY be left back in the server's configuration and be made available to clients.

NOTE: For servers that use an "active" file this means an entry in the form "oldgroup xxx yyy =newgroup" is created.

NOTE: If the old group did not exist, this is considered a local configuration error. Therefore it is the best to correct this error when a mvgroup is received.

If the old group does not exist, the message is ignored unless the new group does not exist either, in which case the new group is created just as for a "newgroup" message.

If both groups exist, the groups MAY be "merged". If this is done, it MUST be done correctly, i.e. implementations MUST take care that the messages in the group being deleted are renumbered accordingly to avoid overwriting articles in one group with those of the other and that crossposted articles don't appear twice. Otherwise, the old group is just deleted.

In all cases, information transported in the "multipart/newsgroupinfo" body part is applied to the new group.

Named articles are taken from the mvgroup message, the new group (if already existent) and the old group in this precedence.

As a special case, the second name, i.e. the one of the new group MAY be omitted. In this case, only the information of the group is updated according to the contained "multipart/newsgroupinfo".

6.6.3.2 Multiple Groups

If the first name ends with the character sequence ".*", the newgroup message requests a whole (sub)hierarchy to be moved. The same procedure as for single groups (section 6.6.3.1) applies to every matched group; however, some systems might be able to optimize the process.

NOTE: For servers that use a file system directory structure to organize message storage, this process can be optimized by renaming the parent directory instead of every group's directory.

To avoid recursion, the new groups' names MUST NEVER match the old groups name pattern; i.e. moving a whole (sub)hierarchy to a subhierarchy of the original hierarchy is explicitly disallowed.

6.6.4 The "checkgroups" Control Message

The "checkgroups" control message contains a list of all valid groups in a complete hierarchy. The "Control:" header has the following format:

```
checkgroup-ctrl = "checkgroups" [ FWS chkscope ] [ FWS chksernr ]
chkscope        = 1*( ["!"] newsgroup-name-part )
chksernr        = "#" 1*DIGIT
```

The chkscope parameter(s) specifies the (sub)hierarchy(s) for which this "checkgroups" message applies.

6.6.4.1 Example:

```
Control: checkgroups de !de.alt #248
```

NOTE: "Old" software is known to ignore this parameter. Thus a "checkgroups" message SHOULD also contain the groups of other subhierarchies the sender is not responsible for. "New" software MUST ignore groups which don't fall into the scope of the "checkgroups" message.

If no scope for the checkgroups message is given, it applies to all hierarchies for which group statements appear in the message.

"Checkgroups" messages MAY also contain a serial number, which can be any positive integer (i.e. just numbered or the date in YYYYMMDD). It SHOULD increase by an arbitrary value with every change to the group list and MUST NOT ever decrease.

NOTE: This was added to circumvent security problems in situations where the Date header can not be signed.

The body of the message is an "application/newscheckgroups" part containing the list of ALL valid groups (and MAYbe deletion confirmations) for the specified hierarchies.

6.6.5 application/newscheckgroups

The "application/newscheckgroups" body part contains a complete list of all newsgroups in a top level hierarchy, their description lines and moderation status.

The MIME content type definition of "application/newscheckgroups:" is:

```
MIME type name:           application
MIME subtype name:        newscheckgroups
Optional parameters:      none
Encoding considerations:  "7bit" or "8bit" is sufficient and
                          MUST be used to maintain compatibility.
                          Note that the descriptions may use [MIME3].
Security considerations:  to be added
```

The content of the "application/newscheckgroups" body part is defined as:

```
checkgroups-body = *( invalidation CRLF ) 1*( valid-group CRLF )
invalidation     = "!" groupname *( "," *WSP groupname )
valid-group      = description-line
description-line      ; see section 6.6.1.2
```

The "application/newscheckgroups" content type is used in conjunction with the "checkgroups" control message (section 6.6.1.3.1).

6.6.5.1 Examples

A "newgroup" with bilingual charter and policy information:

```
From: admin@example.invalid (example.all Administrator)
Newsgroups: example.admin.groups,example.admin.announce
Date: 27 Feb 1997 12:50:22 +14:00 (EST)
Subject: Group example.admin.info created.
Approved: admin@example.invalid
Control: newgroup example.admin.info moderated
Message-ID: <newgroup-example.admin.info-19970227@example.invalid>
Content-Type: multipart/newsgroupinfo; boundary="nxtprt"
Content-Transfer-Encoding: 8bit

This is a MIME control message.
--nxtprt
Content-Type: application/newsgroupinfo

For your newsgroups file:
example.admin.info Information on the example.* hierarchy <info@news.org>
(Moderated)

--nxtprt
Content-Type: multipart/alternative ;
differences = content-language ;
boundary = nxtlang
Article-Name: example.admin.info: charter

--nxtlang
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
```

```
Content-Language: en

The group example.admin.info contains regularly posted information on
the example.* hierarchy.
--nxtlang
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 8bit
Content-Language: de

Die Gruppe example.admin.info enthõlt regelmõ~Kig versandte
Informationen ³ber die example.*-Hierarchie.
--nxtlang--
--nxtprt--
```

plain "rmgroup":

```
From: admin@example.invalid (example.all Administrator)
Newsgroups: example.admin.groups, example.admin.announce
Date: 4 Jul 1997 22:04 +02:00 (PST)
Subject: Deletion of example.admin.obsolete
Message-ID: <rmgroup-example.admin.obsolete-19970730@example.invalid>
Approved: admin@example.invalid
Control: rmgroup example.admin.obsolete

The group example.admin.obsolete is obsolete. Please remove it from
your system.
```

plain "mvgroup":

```
From: admin@example.invalid (example.all Administrator)
Newsgroups: example.admin.groups, example.admin.announce
Date: 30 Jul 1997 22:04 +02:00 (CEST)
Subject: Moving example.oldgroup to example.newgroup
Message-ID: <mvgroup-example.oldgroup-19970730@example.invalid>
Approved: admin@example.invalid
Control: mvgroup example.oldgroup example.newgroup
Content-Type: multipart/newsgroupinfo; boundary=nxt

--nxt
Content-Type: application/newgroupinfo

For your newsgroups file:
example.newgroup The new replacement group.
--nxt

The group example.oldgroup is replaced by example.newgroup.
Please update your configuration.
--nxt--
```

more complex "mvgroup" for a whole hierarchy:

The charter of the group example.talk.jokes contained a reference to example.talk.jokes.d, which is also being moved. So the charter is updated.

```
From: admin@example.invalid (example.all Administrator)
Newsgroups: example.admin.groups, example.admin.announce
Date: 30 Jul 1997 22:04 +02:00 (PST)
Subject: Deletion of example.admin.obsolete
Message-ID: <mvgroup-example.talk-19970730@example.invalid>
Approved: admin@example.invalid
Control: mvgroup example.talk.* example.conversation
Content-Type: multipart/newsgroupinfo; boundary=nxt; chartas=1

--nxt
Content-Type: application/newgroupinfo

For your newsgroups file:
example.conversation.boring Boring conversations.
example.conversation.interesting Interesting conversations.
example.conversation.jokes Jokes and funny stuff.
example.conversation.jokes.d Discussion about example.conversation.jokes.

Article-Name: example.conversation.jokes: charter

This group is to publish jokes and other funny stuff.
Discussions about the articles posted here should be redirected
to example.conversation.jokes.d; adding a Followup-to: header
is recommended.
--nxt--
```

6.6.6 Cancel

The cancel message requests that one or more target articles be "canceled" ie be withdrawn from circulation or access. This message MAY be issued by entities which processed the target article(s) while it was still a proto-article (ie posters, posting agents, moderators and injecting agent. See also Gateways[2.1]). Other entities MUST NOT use this method to remove articles.

NOTE: A separate method for other entities to cancel articles will be defined in a later draft.

```
      cancel-arguments  = 1*( message-id CFWS )
      cancel-body       = body
```

The argument(s) identify the article(s) to be cancelled, by message-id. The body SHOULD contain an indication of why the cancellation was requested. The cancel message SHOULD be posted to the same newsgroup(s), with the same distribution(s), as the article(s) it is attempting to cancel.

In order for a cancel message to remove an article either:

1. The mailing addresses from the From line of the cancel message and the target article match and the target article is otherwise unauthenticated.

2. At least one authentication method of the target article MUST be matched by the cancel message plus the mailing addresses from the From line of the cancel message and the target article MAY match.

NOTE: The Sender, From or Approved headers MUST NOT be used as an "authentication method" within the meaning of the previous paragraph. If the above conditions are satisfied then the relaying or serving agent SHOULD delete the target article completely and immediately (or at the minimum make the article unavailable for relaying or serving) and also SHOULD reject any copies of this article that appear. See also section 7 on duties of Serving and Relaying agents.

6.6.7 ihave, sendme

The ihave and sendme control messages implement a crude batched predecessor of the NNTP [rrr] protocol. They are largely obsolete in the Internet, but still see use in the UUCP environment, especially for backup feeds that normally are active only when a primary feed path has failed.

NOTE: The ihave and sendme messages defined here have ABSOLUTELY NOTHING TO DO WITH NNTP, despite similarities of terminology.

The two messages share the same syntax:

```
       ihave-arguments   = *( message-id space ) relayer-name
       sendme-arguments  = ihave-arguments
       ihave-body        = *( message-id CRLF )
       sendme-body       = ihave-body
```

Message IDs MUST appear in either the arguments or the body, but NOT both. Relayers SHOULD generate the form putting message IDs in the body, but the other form MUST be supported for backward compatibility.

The ihave message states that the named relaying agent has received articles with the specified message IDs, which may be of interest to the relaying agents receiving the ihave message.

The sendme message requests that the agent receiving it send the articles having the specified message IDs to the named relaying agent.

These control messages are normally sent essentially as point-to-point messages, by using "to." newsgroups (see section 5.5.1) that are sent only to the relaying agent the messages are intended for. The two relaying agents MUST be neighbors, exchanging news directly with each other. Each relaying agent advertises its new arrivals to the other using ihave messages, and each uses sendme messages to request the articles it lacks.

To reduce overhead, ihave and sendme messages SHOULD be sent relatively infrequently and SHOULD contain reasonable numbers of message IDs. If ihave and sendme are being used to implement a backup feed, it may be desirable to insert a delay between reception of an ihave and generation of a sendme, so that a slightly slow primary feed will not cause large numbers of articles to be requested unnecessarily via sendme.

6.6.8 Obsolete control messages.

The following forms of control messages are declared obsolete by this document:

```
sendsys
version
whogets
senduuname
```

6.7. Distribution

The Distribution header specifies geographical or organizational limits to an article's propagation:

```
Distribution-content = distribution *( dist-delim distribution)
dist-delim = ","
distribution = positive-distribution / negative-distribution
positive-distribution = *FWS distribution-name *FWS
negative-distribution = *FWS "!" distribution-name *FWS
distribution-name = 1*letter
```

[That is more restrictive than Henry, omitting '+', '-' and '_', but more liberal in allowing uppercase letters, which in fact are commonly used, and in not specifying any 14 character limit.]

A distribution is case-insensitive (i.e. "US", "Us" and "us" all specify the same distribution). In the absence of a Distribution header, the default Distribution-content is "world". However, "world" SHOULD NOT be explicitly mentioned unless a negative-distribution is also present, as in Distribution: world, !us "All" MUST NOT be used as a distribution-name.

Articles MUST NOT be passed between relaying agents unless the sending agent has been configured to supply and the receiving agent has requested to receive BOTH of (a) at least one of the newsgroups in the article's Newsgroups header, and (b) at least one of the positive-distributions in the article's Distribution header and none of the negative-distributions. Exceptionally, ALL relaying agents are deemed willing to supply or accept the distribution "world", and NO relaying agent should supply or accept the distribution "local".

Posting agents SHOULD NOT provide a default Distribution header without giving the poster an opportunity to override it. Followup agents SHOULD initially supply the same Distribution header as found in the precursor.

All the two-letter country names (e.g. "us") commonly used as top-level domain names may be used as distributions, but the common non-country top-level domain names (such as "edu" and "com") are NOT distributions, moreover top-level newsgroup-names (such as "comp" and "soc") are NOT distributions. Apart from the above, distribution-names are a matter for negotiation between the relaying agents or cooperating subnets involved.

6.8. Keywords

The Keywords field contains a comma separated list of important words and phrases intended to describe some aspect of the content of the article. The format of the Keywords header is defined in the Message Format Standard [MESSFOR] .

NOTE: The list is comma seperated NOT space seperated.

6.9. Summary

The Summary header content is a short phrase summarizing the article's content.

```
  summary-content  =  non-blank-text CRLF
  non-blank-text = 1*(FWS text)
```

The summary SHOULD be terse. Authors SHOULD avoid trying to cram their entire article into the headers; even the simplest query usually benefits from a sentence or two of elaboration and context, and not all reading agents display all headers. On the other hand the summary should give more detail than the Subject.

6.10. Approved

The Approved header content indicates the mailing addresses (and possibly the full names) of the persons or entities approving the article for posting:

```
Approved-content = From-content
```

An Approved header is required in all postings to moderated newsgroups. If this header is not present then relaying and serving agents MUST reject the article.

An Approved header is also required in certain control messages, to reduce the probability of accidental posting of same; see the relevant parts of section 6.6.

Please see section 7.1 on how injecting agents should treat posts to moderated groups that do not contain this header.

6.11 Lines

The Lines header content indicates the number of lines in the body of the article:

```
Lines-content = 1*digit
```

The line count includes all body lines, including the signature if any, including empty lines (if any) at beginning or end of the body. (The single empty separator line between the headers and the body is not part of the body) . The "body" here is the body as found in the posted article as transmitted by the posting agent.

Reading agents SHOULD NOT rely on the presence of this header, since it is optional (and some posting agents do not supply it). They MUST NOT rely on it being precise, since it frequently is not.

6.12. Xref

The Xref header content indicates where an article was filed by the last server to process it:

```
     Xref-content = server 1*( CFWS location )
     server = server-name
     location = newsgroup-name ":" article-locator
     article-locator  = 1*
```

The serving agent's name is included so that software can determine which serving agent generated the header. The locations specify what newsgroups the article was filed under (which may differ from those in the Newsgroups header) and where it was filed under them. The exact form of an article locator is implementation-specific.

NOTE: The traditional form of an article locator is a decimal number, with articles in each newsgroup numbered consecutively starting from 1. NNTP demands that such a model be provided, and there may be other software which expects it, but it seems desirable to permit flexibility for unorthodox implementations.

An agent inserting an Xref header into an article MUST delete any previous Xref header(s). A relaying agent MUST only create and/or relay an Xref header if it correct on all the receiving agents the article is forwarded to. Serving agents SHOULD insert this header unless the information in it (apart from the serving name) is correct in which case it should be left unchanged.

An agent MUST use the same name in Xref headers as it uses in Path headers.

6.13. Organization

The Organization header content is a short phrase identifying the author's organization:

```
organization-content = nonblank-text CRLF
```

NOTE: Posting and injection agents are discouraged from providing a default value for this header unless it is acceptable to all posters using these agents. Unless this header contains useful information (including some indication of the authors physical location) posters are discouraged from including it.

6.14. User-Agent

The User-Agent header contains information about the user agent (typically a newsreader) generating the article. This is for statistical purposes and tracing of standards violations to specific software needing correction. Although OPTIONAL, user agents SHOULD include this header with the articles they generate.

The field MAY contain multiple product tokens and comments identifying the agent and any subproducts which form a significant part of the user agent such as external agents used for message composition, separated injecting agents (such as those used by offline newsreaders), and significant libraries that are part of such agents. The products are listed in order of their significance for identifying the application, not necessarily in chronological order of handling prior to injection. Injecting agents MAY include product information for servers (such as INN/1.7.2), but servers MUST NOT generate or modify this header to list themselves.

User-Agent MUST NOT be modified after injection, but MAY be stripped or have its contents replaced prior to re-injection by another user agent such as an anonymizing gateway.

```
User-Agent = "User-Agent:" SP User-Agent-content
User-Agent-content = product *(CFWS product) [CFWS]
```

At least one product MUST be present. The first token MUST NOT be a comment. Comments relate to the previously named product, not the product following it.

```
product = token ["/" product-version] product-version = token
```

Product tokens should be short and to the point -- they MUST NOT be used for information beyond the canonical name of the product and it's version. Although any token character MAY appear in a product-version, this token SHOULD be used only for a version identifier (i.e., successive versions of the same product SHOULD differ only in the product-version portion of the product value). Product tokens MUST identify products.

NOTE: Variations from RFC 1945:

1. product token is required and MUST be first,

2. use of other text in the syntactic usage of the product token which is not a token is forbidden,

3. comment allows quoted-pair,

4. "{" and "}" are allowed in token (product and product-version) in news,

5. octets from character sets other than ASCII are allowed.

NOTE: Comments should be restricted to information regarding the product named to their left such as platform information and should be concise. Use as an advertising medium (in the mundane sense) is discouraged.

Recipients of header field TEXT containing octets outside the US-ASCII character set may assume that they represent UTF-8 characters.

NOTE: Variation from RFC 1945: UTF-8 replaced ISO-8859-1 as charset assumption.

6.14.1 Examples:

```
User-Agent: tin/1.2-PL2
User-Agent: tin/1.3-950621beta-PL0 (Unix)
User-Agent: tin/unoff-1.3-BETA-970813 (UNIX) (Linux/2.0.30 (i486))
User-Agent: tin/pre-1.4-971106 (UNIX) (Linux/2.0.30 (i486))
User-Agent: Mozilla/4.02b7 (X11; I; en; HP-UX B.10.20 9000/712)
User-Agent: Microsoft-Internet-News/4.70.1161
User-Agent: Gnus/5.4.64 XEmacs/20.3beta17 ("Bucharest")
User-Agent: Pluto/1.05h (RISC-OS/3.1) NewsHound/1.30
User-Agent: inn/1.7.2
User-Agent: inews
User-Agent: telnet
```

NOTE: Some current web proxy applications append their product information to the list in the User-Agent field. This is not recommended on the web and is forbidden for news, since it makes machine interpretation of these fields ambiguous. User-Agent is not intended to be a total audit trail of what software has handled the article.

NOTE: Some existing web clients fail to restrict themselves to the product token syntax within the User-Agent field when using this header on the web. Such abuses are forbidden for news.

NOTE: This header supersedes the role performed redundantly by "X-" headers such as X-Newsreader, X-Mailer, X-Posting-Agent, X-Http-User-Agent, and other headers previously used on USENET for this purpose. Use of these "X-" headers SHOULD be discontinued in favor of the single, standard User-Agent header which is to be used freely both in news and mail.

NOTE: There are slight changes to the original HTTP defined format to the User-Agent header as noted, but headers in strict, common-sense compliance with RFC 1945 are compliant to this specification. The syntax from RFC 1945 is preferred, including the requirement that products and comments be separated by a space.

6.15 MIME headers

6.15.1 Syntax

The following headers, as defined within [RFC 2045] and its extensions, may be used within articles conforming to this document.

MIME-Version:
Content-Type:
Content-Transfer-Encoding:
Content-ID:
Content-Description:
Content-Disposition:
Content-MD5:

Insofar as the syntax for these headers as given in [RFC 2045] does not specify precisely where whitespace and comments may occur (whether in the form of WS, FWS or CFWS), the usage defined in this Standard, and failing that in [MESSFOR], and failing that in [RFC 822] MUST be followed. In particular, there MUST NOT be any WS between a header-name and the following colon and there MUST be a SPACE following that colon.

The meaning of the various MIME headers is as defined in [RFC 2045] and [RFC 2046], and in extensions registered in accordance with [RFC 2048]. However, their usage is restricted as described in the following sections.

6.15.2 Content-Transfer-Encoding

Posting agents SHOULD specify "Content-Transfer-Encoding: 8bit" for all articles not written in pure ASCII or whose content type implies binary. They MAY use "8bit" encoding even when "7bit" encoding would have sufficed. They SHOULD specify "base64" when the content type implies binary (i.e. content intended for machine, rather than human, consumption).

Posting agents SHOULD NOT specify encoding "quoted-printable", but reading agents MUST interpret that encoding correctly. Encoding "binary" MUST NOT be used because this Standard does not mandate a transport mechanism that could support it (exceptions might be made in closed networks with alternative transport arrangements).

Injecting and relaying agents MUST NOT change the encoding of articles passed to them. Gateways SHOULD ONLY change the encoding if absolutely necessary.

6.15.3 Content-Type

Network news is primarily a means of sharing textual information amongst a wide audience in a timely manner. The network is largely self regulating and operates by the consensus of its membership rather than by the dictate of any central authority (indeed, this lack of centralised control is seen as one of the overall strengths of the system). There are practices which, whilst being technically unexceptionable, are politically undesirable (or contrary to established "netiquette").

Insofar as there exist authorities empowered (by common consent or otherwise) to define what is and is not proper in various hierarchies or newsgroups or cooperating subnets, those authorities ought to establish, by means of rules, guidelines, charters or whatever else, the practices considered acceptable within their domains. In particular they ought to establish which of the more exotic content types are likely to be inappropriate. In the absence of such specific guidance, the following default recommendations are offered as an indication of best practice at the present time.

6.15.3.1 Text

"Content-Type: text/plain" is the expected type for any news article. Attention is drawn to the recommendations and limits on line lengths set out in section 4.6. Indeed, in any "text" content type the lines as transmitted (i.e. including any formatting instructions) ought to observe the recommendations set out in that section for the benefit of readers who can only see it in its transmitted form.

While "Content-Type: text/enriched" [RFC 1896] can be considered acceptable in news articles, "Content-Type: text/HTML" is not appropriate since it relies on protocols currently unavailable to many reading agents.

6.15.3.2 Application

Generally speaking, the application content types are inappropriate for use outside of specialised newsgroups and subnets, especially where vendor-specific application subtypes are involved.

"Application/octet-stream" is only appropriate in newsgroups where binaries are customarily accepted.

6.15.3.4 Image, Audio and Video

Likewise, these content types are only appropriate in newsgroups where binaries are customarily accepted.

6.15.3.5 Multipart

"Content-Type: multipart/mixed" (also "multipart/parallel") may be used freely in news articles.

The use of "Content-Type: multipart/alternative" is deprecated (on account of the extra bandwidth consumed and the difficulty of quoting in followups).

"Content-Type: multipart/digest" is recommended for any article composed of multiple messages more conveniently viewed as separate entities. The "boundary" should be composed of 28 hyphens (ASCII 45) (which makes each boundary delimiter 30 hyphens, or 32 for the final one) so as to accord with current practice for digests [RFC 1153].

"Content-Type: multipart/signed" [RFC 1847, RFC 2015] is the preferred method for the bodies of news articles that are to be digitally signed. However, encryption (as in "multipart/encrypted") is unlikely to be appropriate in a medium normally directed at such a wide readership.

6.15.3.6 Message

The Content Types "message/rfc822" and "message/news" are unlikely to be of much use within news articles, but attention is drawn to the benefits of using "message/news" in gatewaying from mail to news and vice versa. In particular, news articles mailed to moderators SHOULD be totally encapsulated within an email message using "message/news". Moreover, use of "message/news" in conjunction with a suitable Transfer Encoding forms a convenient way of "tunnelling" a news article through a transport medium that does not support 8bit characters.

[This paragraph needs further work. Both message/news and application/news-transmission are recognised by IANA, but the distinction between them is not clear and their present definitions are out of date and omit to state that base 64 etc encodings are permitted (RFC 2046 is silent on that issue). We should take the opportunity to rewrite those specifications and include them (or at least one of them) in the present standard.]

"Content-Type: message/partial" MAY be used to split a long news article into several smaller ones, but this usage is deprecated on the grounds that modern transport agents should have no difficulty in handling articles of arbitrary length. If this feature is used, then the "id" parameter should be in the form of a unique message-id (but different from the Message-ID of any of the partial articles). The second and subsequent partial articles should contain References headers referring to all the previous parts (note that these headers will be discarded upon reassembly of the parts). Contrary to

the requirements specified in [RFC 2046], the
Transfer-Encoding should be set to 8bit at least in each part
that requires it.

"Content-Type: message/external-body" could be appropriate for
texts which it would be uneconomic (in view of the likely
reader- ship) to distribute to the entire network.

6.15.3.7 Character Sets

In principle, any character set may be specified in the
"charset=" parameter of a content type. However, character
sets other than "us-ascii", "iso-8859-1" (and the
corresponding parts of UTF-8) ought only to be used in
hierarchies where the language customarily used so required
(and whose readers could be expected to possess agents capable
of displaying them).

6.15.4 MIME within headers

Since the headers of news articles are expected to use the
UTF-8 character set, they SHOULD NOT normally be encoded using
the MIME mechanism defined in RFC-2047 [RFC-2047].
Nevertheless, reading agents SHOULD support that usage.

It is to be noted, however, that RFC-2047 encoding would be
required were a news article to be transmitted as a mail
message without first encapsulating in as a "message/news" as
suggested above.

6.15. Supersedes / Replaces

These two headers take a list of message-ids (msgid-list) that
the current article is expected to replace or supersede. All
listed articles MUST be treated as though a "cancel" control
message had arrived for the message, except as detailed below.

These headers are essentially synonyms, with a change in
behavior - Replaces uses the old article's message-id for
the more recently arrived article, rather than creating a
new article.

The Supersedes header content specifies articles to be
cancelled on arrival of this one:

```
     Supersedes-content = message-id *( FWS message-id )
```

NOTE: There is no "c" in "Supersedes".

Older software supported only Supersedes, and with only one
Message-ID. Until Multi-Super-Date, software SHOULD generate
Supersedes with only one Message-ID, and cancel control

messages SHOULD be issued if needed for other IDs.

If the header is "Replaces" the new successor article SHOULD effectively over-write the predecessor(s) so that any attempt to read them shows the successor. Newsreaders should not show the article as an "unread" article unless the replaced articles were themselves all unread. A Replacement is considered a minor change, unworthy of being brought to the attention of a person who read one of the predecessors. Newsreaders and database systems MAY provide access to predecessors of articles if they wish, but this should not be part of the course of normal newsreading, and is in fact discouraged.

Systems MAY treat Replaces as a synonym for Supersedes, if they do not implement the semantics of Replaces.

If the header is "Supersedes" then the old articles SHOULD simply be deleted, as in a cancel, and the new article inserted into the system like any new article.

Attempts to fetch a replaced or superseded article either by number or by Message-ID SHOULD retrieve instead the most recent successor. Some indication that a newer version than was asked for has been delivered MAY be provided. It is particularly encouraged that NNTP servers implement delivery of successor upon requests by message-IDs so that WWW "news:" and "msg:" URLs continue to work even when an article has a successor.

It is expected that "Replaces" will become the common header for routine article changes and corrections, with Supersedes used for periodic postings (possibly every N periodic postings) or updates that make major changes to an article.

As with a cancel, systems MUST NOT delete or replace articles unless the poster of the successor is authorized to cancel the predecessor.

6.15.1 Message-ID version numbers chain procedure.

NOTE: Sections 6.15.1 - 6.15.4 may be published as a separate recommendations document.

Tools superseding or replacing messages should arrange so that the Message-ID of a replacement follows the following set of rules, generating what are known as "version-number" Message-IDs.

1. If the Local-Part of the predecessor's Message-ID ends in "%v=<n>", where <n> is an integer version number, the new message-ID should replace the <n> with the integer <n+1>.

Example:
Message-ID <foo%v=3@example.invalid> is replaced by
<foo%v=4@example.invalid>.

2. If the Local-Part of the predecessor's Message-ID does not end in "%v=<n>", then the string "%v=1" should be appended to the Local-Part to generate the successor Message-ID.

Example:
Message-ID <foo@example.invalid> is replaced by
<foo%v=1@example.invalid>.

6.15.2 Implementation and Use Note

Typically a news database will store a "pointer" of some sort between replaced/superseded articles and their immediate successor or most recent successor. Such pointers may be expired along with other records in a news system's message-id lookup database. In addition, if a "version-number" Message-ID is found, and the "root" version (without the "%v=" tag, or with a "%v=0" tag) is not present on the server, a pointer from that root to the most recent successor SHOULD also be stored, and kept so long as there is a current successor in the system. (Systems should check for both root forms, trying the "%v=0" form first, and the tagless form 2nd.)

Thus when a request for an article comes in that is not present (due to superseding or replacement) a check can be made for a pointer record for that Message-ID, or failing that, if the ID has a version-number, for a pointer record for the root versionless ID. Such pointers can be followed to the most recent successor.

6.15.3 Transition

Prior to Multi-Super-Date, a message may contain both a Replaces field and a Supersedes field. This should be treated as a Replaces, with the Supersedes added to assure that older systems still at least remove the predecessor.

6.15.4 Replaced-by

This header takes a message-id as argument.

Prior to Multi-Super-Date, if there is a need to Supersede by use of a simple Cancel control message (due to inability to use multiple IDs in the Supersedes header) then such control messages may contain a "Replaced-by" header indicating the Message-ID of the successor the message that was cancelled. Systems maintaining pointers from predecessors to successors should use this record to update their pointers.

Note this header goes only on the cancel control message, not the successor. The successor should have a Replaces and/or Supersedes listing the most immediate predecessor.

6.15.5.1 Examples

The first edition of an FAQ is posted with a Message-ID of the form: <examplegroup-faq@example.invalid>. The next version, a week later, has:

```
Message-ID:     <examplegroup-faq%v=1@example.invalid>
Supersedes:     <examplegroup-faq@example.invalid>
```

The next one, another week later has:

```
Message-ID:     <examplegroup-faq%v=2@example.invalid>
Supersedes:     <examplegroup-faq%v=1@example.invalid>
<examplegroup-faq@faqsite.com>
```

The next one, another week later has:

```
Message-ID:     <examplegroup-faq%v=3@example.invalid>
Supersedes:     <examplegroup-faq%v=2@example.invalid>
<examplegroup-faq@example.invalid>
```

Note that the long spacing between issues means the multi-entry Supersedes is there primarily to preserve pointer records at sites not using the version-number system for message-ids.

Under the above, requests for the root (original) message-ID will return the most recent FAQ. On systems using the version-number system (which is optional) requests for any Message-ID in the chain will return the most recent, for all time. As such the URL "news:groupname-faq@faqsite.com" will always work, making it suitable to appear in HTML.

6.15.5.2 Example

A user posts a message <myuniquepart@mysite.com> to the net. She notices a typo, and 2 minutes later, posts with:

```
Message-ID:     <myuniquepart%v=1@example.invalid>
Replaces:       <myuniquepart@example.invalid>
```

3 minutes later she sees another typo, and posts:

```
Message-ID:     <myuniquepart%v=2@example.invalid>
Replaces:       <myuniquepart%v=1@example.invalid>
<myuniquepart@example.invalid>
```

The two bad versions will be replaced with the 3rd, even if a site never sees the 2nd due to batching or feed problems, and requests for the original will return the 3rd.

During transition, she adds a Supersedes header to the 3rd message, with the first (direct predecessor) ID. She issues a Cancel message as well:

```
Control: cancel <myuniquepart@example.invalid> Replaced-by:
<myuniquepart%v=2@example.invalid>
```

6.15.6 Dates

Multi-Super-Date ... in one year. (1036-spencer required multiple-ID supersedes, so by now just about everybody should already support it, is this true?) "Replaces" active -- whatever date we are putting for general compliance with this spec by news database systems.

6.15.7 Issues

No syntax for the internals of message-ids has been declared on the net. However, there is no harm if a conforming message-id matches the syntax. The syntax has been designed so that additional flags may be added to a message-id if desired, in a general "%keyword=value" form prior to the at-sign.

Permanent message-ids as created by this system may even be implemented by smart NNTP servers which fetch old messages from other servers, increasing the availability of USENET messages considerably.

Unfortunately, it will be some time until any new feature is widely deployed.

6.16 Archive

This optional header is a signal to automatic archival agents on whether this article is available for long-term storage. Agents which see "Archive: no" MUST NOT keep the article past the Expires date. Any other text indicates conditions that an agent SHOULD follow in order to archive the article.

```
Archive-content = "no" | CFWS
```

6.17. Obsolete Headers

Persons writing new agents SHOULD ignore any former meanings of these headers.

Also-Control
See-Also

Article-Names
Article-Updates

7. Duties of Various Agents

The following section sets out the duties of various Agents involved in the creation, relaying and serving of Usenet articles.

Agents which write to the Path header MUST conform to RFC2142 with respect to contact addresses especially the "usenet" and "abuse" addresses.

7.1 Duties of an Injecting Agent.

An injection agent is responsible for taking a proto-article from a posting agent and either forwarding it to a moderator of injecting it into the relaying system for access by readers.

As such a Injecting Agent is considered responsible for ensuring that any article it injects conform with the policies and rules of this document and any newsgroups that an article is posted to.

To this end injection agents MAY cancel articles which they have previously injected.

7.1.1 Proto-articles.

A proto-article is one that is created by a posting agent and has not been injected into the news system by an injecting agent. Only one copy of a proto-article MUST exist. A proto-article has the same format as a normal article except that some of the compulsory headers MAY be missing. A proto-injected article MAY have the following headers missing: "Message-Id: " , "Date: " and "Path: " . These header MUST not contain invalid values, they MUST either be correct or not present at all.

A proto-article MUST NOT contain the "!" or "%" character in the Path header.

Proto-articles SHOULD NOT contain the Originator-Info header. See reference [draft-newman-msgheader-originfo-x.txt] on this header for more information.

7.1.2 Procedure followed by Injecting Agents.

A injecting agent receives proto-articles from posting and followup agents. It verifies them, adds headers where required and then either forwards them to a moderator or injects them by passing them to serving or relaying agents. An injecting agent SHOULD only accept articles from trusted agents.

An injecting agent MAY reject articles in which headers contain "forged" email addresses, that is, addresses which are not valid for the known source, and do not end in ".invalid".

If an injecting agent receives an otherwise valid article that has already been injected it SHOULD either act as if it is a relaying agent or pass the article on to a relaying agent completely unaltered. It MUST NOT forward an already injected article to a moderator. Articles SHOULD NOT be injected twice.

An injecting agent accepts a proto-article checks it and does one of the following:

(a) If the article is invalid, incorrectly formatted or unacceptable due to site policy the posting agent MUST be informed (such as via a [NNTP] 44x response code) that posting has failed and the article MUST NOT be injected nor forwarded to a moderator.

(b) If the Newsgroups line contains one or more moderated groups and the article does NOT contain an Approved header then the injecting agent MUST forward the article to the moderator of the first (leftmost) moderated group listed in the Newsgroups line via email. The injecting agent MUST also add headers as detailed below.

(c) If the proto-article is not posted to any moderated newsgroups or the Approved header is correctly present then the injecting agent should convert the proto-article to an injected article (see below) and forwarded it to one or more relaying or serving agents.

7.1.3 Headers added by Injecting Agents.

When an injecting agent forwards and article to a moderator or injects it it MUST do the do the following:

The message-id and Date headers (and their content) MUST be added if not already present. The Path header MUST be correctly added if the article is being injected but SHOULD NOT be added if it is being forwarded to a moderator.

If the Originator-Info header is already present in the proto-article then it MUST be removed if incorrect and a correct one MAY added.

The Injecting Agent MUST verify the poster in some way. The Path header (section 5.6) MUST be correctly used and some other secure standard method (such as the Originator-info header) MAY be used.

The Injecting Agent MAY add other headers not listed in this draft but MUST NOT alter, delete or reorder any headers already present in the article except the Originator-Info header (see above). The Injecting Agent MUST NOT alter the body of the article in any way.

7.2 Duties of a Relaying Agent

A relaying Agent accepts injected articles from injecting and other relaying agents and passes them on to relaying agents or serving agents according to mutually agreed policy. Relaying Agents SHOULD only accept articles from trusted agents.

A relaying agent MAY reject articles in which headers contain "forged" email addresses, that is, addresses which are not valid for the known source, and do not end in ".invalid".

A relaying agent MUST perform checks on an article to ensure it complies with this standard. If the article is invalid, unwanted (see below) or unacceptable due to site policy the agent that passed the article to the relaying agent SHOULD be informed (such as via a [NNTP] 43x response code) that relaying failed. In order to prevent a large number of error messages being sent to one location relaying agent MUST NOT inform any other external entity that an article was not relayed UNLESS that external entity has specificly requested that it be informed of these errors.

In order to prevent overloading relaying agents SHOULD NOT query an external entity (such as a key-server) in order to verify an article.

When an article is received the relaying agent MUST verify the previous entry in this header and add their own entry(s) according to the syntax defined in the Path section 5.6. Relaying agents MUST NOT alter, deleted or rearrange any part of an article expect for the Path and Xref Headers.

Article which match mutually agreed criteria should be passed onto neighboring relaying and serving agents.

NOTE: It is usual for relaying and serving agents to restrict the Newsgroups, Distributions, age and size of articles they wish to receive.

7.2.1 Unwanted and Invalid articles

Relaying Agents MUST reject all articles that do not have all mandatory headers present with legal contents or which have illegal contents in optional headers.

Relaying Agents SHOULD reject any articles that have already been sent to it (a database message-ids of recent messages is usually kept and matched against) or which are too old (from the Date header) for it to determine if they have already been sent to it. Relaying Agents SHOULD NOT forward articles to sites whose path-identity is already in the Path header.

Relaying Agents SHOULD also reject any articles that have been Canceled, Superseded or Replaced by their author or another trusted entity.

7.3 Duties of a Serving Agent

A Serving Agent takes an article from a relaying or injecting agent and files it in a "news database" . It also provides an interface for reading agents to access the news database. This database is normally indexed by newsgroup with with articles in each newsgroup numbered consecutively starting from 1. See [NNTP] for more information on this format.

NOTE: Control messages are usually filed in the separate pseudo-newsgroup "control" or a pseudo-newsgroup in a hierarchy under "control." (ie "control.cancel") . Serving Agents SHOULD do this if they serve articles via NNTP.

Serving Agents are encouraged to only allow access to trusted reading agents.

Serving Agents SHOULD generate a correct Xref header for crossposted articles and MUST prepend a correct path-identity into the Path header of all articles.

7.3.1 Unwanted articles

Serving Agents MUST reject all articles that do not have all mandatory headers present with legal contents or which have illegal contents in optional headers.

Serving Agents SHOULD reject any articles that have already been sent to it (a database message-ids of recent messages is usually kept and matched against) or which are too old (from the Date header) for it to determine if they have already been sent to it.

Serving Agents SHOULD also reject any articles that have been Canceled, Superseded or Replaced by their author or another trusted entity and delete any of these articles that they already have in their news database.

7.4 Duties of a Posting Agent.

A posting agent is used to assist the poster in creating a valid proto-article and forwarding it to an injecting agent.

Postings Agents SHOULD ensure that proto-articles they create are valid usenet articles according to the standards of this document and other policies.

Posting agents meant for use by ordinary posters SHOULD reject any attempt to post an article which cancels, Supersedes or Replaces another article if the target article not by the poster.

7.5 Duties of a Followup Agent

A followup Agent is a special case of a Posting Agent and as such is bound by all the Posting Agent's requirements plus additional ones. Followup Agents MUST create valid followups, Followups have additional requirements from normal articles for the syntax of the References and Subject headers and the body format.

Followup Agents MUST by default follow the FollowUp-To header when deciding which newsgroups a followup is posted to, however the poster MAY override the default if they wish.

Followup Agents MUST NOT attempt to send email to any address ending in ".invalid".

Followup Agents SHOULD NOT email copies of the followup to the author of the precursor (or any other person) unless this has been explicitly requested.

7.6 Duties of a Gateway

NOT DONE

8. Propagation and Processing

Most aspects of news propagation and processing are implementation-specific.   The basic propagation algorithms, and certain details of how they are implemented, nevertheless need to be standard.

There are two important principles that news implementors (and administrators) need to keep in mind. The first is the well-known Internet Robustness Principle:

> Be liberal in what you accept, and conservative in what you send.

However, in the case of news there is an even more important principle, derived from a much older code of practice, the

Hippocratic Oath (we will thus call this the Hippocratic
Principle):

First, do no harm.

It is VITAL to realize that decisions which might be merely
suboptimal in a smaller context can become devastating
mistakes when amplified by the actions of thousands of
hosts within a few hours.

In the case of gateways, the primary corollary to this is:

Cause no loops.

9. Security And Related Issues

There is no security. Don't fool yourself. USENET is a prime
example of an Internet Adhocratic-Anarchy; that is, an
environment in which trust forms the basis of all agreements.
It works.

Articles which are intended to have restricted distribution
are dependent on the goodwill of every site receiving them.
The "X-No-Archive: yes" header is widely recognized as a
signal to automated archivers not to file an article, but that
cannot be guaranteed.

The Distribution header makes provisions for articles which
should not be propagated beyond a cooperating subnet. The key
security word here is "cooperating". When a machine is not
configured properly, it may become uncooperative and tend to
distribute all articles.

9.1 Attacks

The two categories of attacks that news is most vulnerable to
are Denial-of-Service and exploitations of particular
implementations. Many have argued that "spam", massively
crossposted or reposted articles constitutes a DoS attack in
its own regard. This may be so.

Sending off-topic messages is a matter for individual hierarchies
and newsgroups to control. It is a violation of this DRAFT to
"forge" an email address, that is, to use a valid email address
which you are not entitled to use. All invalid email addresses
used in headers MUST end in the ".invalid" top-level-domain.
This facility is provided primarily for those who wish to remain
anonymous, but do not care to take the additional precautions of
using more sophisticated anonymity measures.

It is possible that legal penalties may apply to sending unsolicited commercial email and/or news articles. Check with your local legal authorities.

10. Security Considerations

Section 9 discusses security considerations.

11. References:

[TEST-TLDS]
Eastlake, D. ; Panitz A. Reserved Top Level DNS Names, draft-ietf-dnsind-test-tlds-xx.txt, May 1998

[ANSI-X3.4] US-ASCII
American National Standard for Information Systems - Coded Character Sets - 7-Bit American National Standard Code for Information Interchange (7-Bit ASCII). ANSI X3.4, 1986.

[ISO-8859]
International Standard - Information Processing - 8-bit Single-Byte Coded Graphic Character Sets - Part 1: Latin alphabet No. 1, ISO 8859-1, 1987. Part 2: Latin alphabet No. 2, ISO 8859-2, 1987. Part 3: Latin alphabet No. 3, ISO 8859-3, 1988. Part 4: Latin alphabet No. 4, ISO 8859-4, 1988. Part 5: Latin/Cyrillic alphabet, ISO 8859-5, 1988. Part 6: Latin/Arabic alphabet, ISO 8859-6, 1987. Part 7: Latin/Greek alphabet, ISO 8859-7, 1987. Part 8: Latin/Hebrew alphabet, ISO 8859-8, 1988. Part 9: Latin alphabet No. 5, ISO 8859-9, 1990. Part 10: Latin alphabet No. 6, ISO 8859-10, 1992.

[ISO-10646]
International Standard - Information technology - Universal Multiple-Octet Coded Character Set (UCS) - Part 1: Architecture and Basic Multilingual Plane. ISO/IEC 10646-1, 1993.

[MESSFOR]

[Originator-Info]
draft-newman-msgheader-originfo-x.txt
chris.newman@innosoft.com

[RFC-822] e-mail message format
Crocker, David H.: Standard for the format of ARPA Internet text messages. RFC 822, 1982-08-13.

[RFC-850] netnews message format (obsolete)

Horton, Mark R.: Standard for interchange of Usenet messages. RFC 850, 1983-06.

[RFC-976] UUCP mail interchange
Horton, Mark R.: UUCP mail interchange format standard. RFC 976, 1986-02.

[RFC-977] NNTP
Kantor, Brian; Lapsley, Phil: Network news transfer protocol - a proposed standard for the stream-based transmission of news. RFC 977, 1986-02.

[RFC-1036] netnews message format
Horton, Mark R.; Adams, R.: Standard for interchange of Usenet messages. RFC 1036, 1987-12.

[RFC-1036BIS] netnews message format (memo)
Spencer, Henry: News article format and transmission. 1994-06-02.

[RFC-1884] IP v6
Hinden, Robert M.; Deering, Stephen E.: IP version 6 addressing architecture. RFC 1884, 1995-12.

[RFC-2045] MIME, part 1
Freed, Ned; Borenstein, Nathaniel S.: Multipurpose Internet mail extensions (MIME), part 1: format of Internet message bodies. RFC 2045, 1996-11.

[RFC-2119] MUST/SHOULD/MAY
Bradner, Scott: Key words for use in RFCs to indicate requirement levels. RFC 2119, 1997-03.

[RFC-2130] character-set memo
Weider, Chris; Preston, Cecilia; Simonsen, Keld; Alvestrand, Harald T.; Atkinson, Randall; Crispin, Mark; Svanberg, Peter: The report of the IAB character set workshop. RFC 2130, 1997-04.

[RFC-2142] standard mailbox names
Crocker, David H.: Mailbox names for common services, roles and functions. RFC 2142, 1997-05.

[RFC-2234] ABNF
Crocker, David H.; Overell, Paul: Augmented BNF for syntax specifications: ABNF. RFC 2234, 1997-11.

[RFC-2279] UTF-8
Yergeau, Francois: UTF-8, a transformation format of ISO 10646. RFC 2279, 1998-01.

[UNICODE] Unicode

The Unicode Consortium: The Unicode Standard - Version 2.0.
Addison-Wesley, 1996.

Author's Address

The Usenet Format Working Group (usenet-format@clari.net)

Deliberations were archived at
http://www.landfield.com/usefor/

Chair: Simon Lyall, simon@darkmere.gen.nz
Editor: Dan Ritter,   dsr@bbn.com

Members (alphabetical):

A. Deckers (Alain.Deckers@man.ac.uk)
Ade Lovett (ade@demon.net)
Andrew Gierth (andrew@erlenstar.demon.co.uk)
Bill Davidsen (davidsen@prodigy.com)
Bill McQuillan (mcquillan@mpa15ab.mv.unisys.com)
Brad Templeton (brad@clari.net)
Brian Hernacki (bhern@netscape.com)
Brian Kelly (bkelly@sulaco.com)
Bryan Ford (baford@sleepless.com)
Buddha Buck (bmbuck@acsu.buffalo.edu)
Charles Lindsey (chl@clw.cs.man.ac.uk)
Chris Newman (chris+ietf-usenet@iosoft.com)
Christian Weisgerber (naddy@mips.rhein-neckar.de)
Christopher Sedore (cmsedore@maxwell.syr.edu)
Claus Andre Farber (lists/usenet-format/clari.net@faerber.muc.de)
Clive D.W. Feather (clive@demon.net)
Curt Welch (curt@kcwc.com)
D. J. Bernstein (djb@koobera.math.uic.edu)
Dave Barr (barr@math.psu.edu)
Dave Hayes (dave@kachina.jetcafe.org)
Dave Mack (dmack@corp.webtv.net)
David C Lawrence (tale@isc.org)
David desJardins (desj@rt.com)
Denis McKeon (DMckeon@swcp.com)
Dirk Nimmich (dirk@roxel.ms.sub.org)
Doug Royer [N6AAW] (dougr@basilisk.Eng.Sun.COM)
Egil Kvaleberg (egil@kvaleberg.no)
Eivind Tagseth (eivindt@multinet.no)
Erik van der Poel (erik@netscape.com)
Erland Sommarskog (sommar@algonet.se)
Evan Champion (evanc@synapse.net)
Fergus Henderson (fjh@cs.mu.oz.au)
Frederic SENIS (fs@caduceus.frmug.org)

Fredric Logren (Fredric.Lonngren@uab.ericsson.se)
Greg Berigan (gberigan@cse.unl.edu)
Harald Alvestrand (Harald.T.Alvestrand@uninett.no)
Heiko Schlichting (heiko@cis.fu-berlin.de)
Heiko W.Rupp (hwr@pilhuhn.de)
Hrvoje Niksic (hniksic@srce.hr)
Ian Davis (iand@fdc.co.uk)
Ian G Batten (I.G.Batten@batten.eu.org)
John Moreno (phenix@interpath.com)
John Stanley (stanley@oce.orst.edu)
Jon Ribbens (jon@oaktree.co.uk)
Jonathan Grobe (grobe@netins.net)
Kai Heingsen (kai@khms.westfalen.de)
Karl Kleinpaste (karl@jprc.com)
Keeth Herron (kherron@campus.mci.net)
Kent Landfield (kent@landfield.com)
Kristian =?ISO-8859-1?Q?K=F6hntopp?= (KRIS@koehntopp.de)
Lars Magne Ingebrigtsen (lmi@gnus.org)
Leonid Yegoshin (egoshin@genesyslab.com)
Mark Hittinger (bugs@freebsd.netcom.com)
Mark Sidell (Mark.Sidell@forteinc.com)
Martin Forssen (maf@math.chalmers.se)
Martin J. Duerst (mduerst@ifi.unizh.ch)
Maurizio Codogno (mau@beatles.cselt.it)
Mick Brown (Mick.Brown@worldnet.att.net)
Mustafa Soysal MS57 (msoysal@mistik.express.net)
Oo Hovers (onno@surfer.xs4all.nl)
Paul Eggert (eggert@twinsun.com)
Paul Overell (richard@pillar.turnpike.com)
Per Abrahamsen (abraham@dina.kvl.dk)
Pete Resnick (presnick@qualcomm.com)
Peter =?ISO-8859-1?Q?Br=FClls?= (pb@Ecce-Terram.DE)
Peter Heirich (peter@heirich.in-berlin.de)
Ralph Babel <rbabel@babylon.pfm-mainz.de>
Richard Clayton (richard@pillar.turnpike.com)
Robert Elz (kre@muari.OZ.AU)
Russ Allbery (rra@stanford.edu)
R. Kelley Cook (kcook@ibm.net)
Seth Breidbart (sethb@panix.com)
Shmuel Metz (shmuel@os2bbs.com)
Simon Fraser (smfr@santafe.edu)
Stan Barber (sob@academ.com)
Sylvan Butler (SBUTLER@hpbs2024.boi.hp.com)
Terje Bless (link@tss.no)
Thomas Roessler (roessler+1036@sobolev.iam.uni-bo.de)
Tim Skirvin (tskirvin@math.uiuc.edu)
Todd Michel McComb (mccomb@best.com)
Tom Hughes (tom@compton.demon.co.uk)
Vera Heinau (heinau@cis.fu-berlin.de)
Wayne Davison (wayne@clari.net)
Wolfgang Schelongowski (skaranyi@xivic.ruhr.de)

Expires 19990101