INTERNET-DRAFT
Usenet Format Working Group

Charles H. Lindsey
University of Manchester
May 2004

News Article Format and Transmission
<draft-ietf-usefor-article-13.txt>

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC 2026.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

Abstract

This Draft is intended as a standards track document, obsoleting RFC 1036, which itself dates from 1987.

This Standard defines the format of Netnews articles and specifies the requirements to be met by software which originates, distributes, stores and displays them.

Since the 1980s, Usenet has grown explosively, and many Internet and non-Internet sites now participate. In addition, the Netnews technology is now in widespread use for other purposes.

Backward compatibility has been a major goal of this endeavour, but where this standard and earlier documents or practices conflict, this standard should be followed. In most such cases, current practice is already compatible with these changes.

[Draft-10 removed all the Internationalization features (i.e. those involving the use of the UTF-8 charset in headers). This is being done so as to facilitate publishing those features, or similar ones, as an experimental protocol at a later stage.

A companion Current Best Practice document [USEAGE], addressing
requirements which are present for Social rather than Normative reasons

is in preparation.

There may well be a further split of the present draft into a syntax/semantics part and a protocol part. Thus this present draft is but an intermediate stage in an ongoing development process.]

[The use of the words "this standard" within this document when referring to itself does not imply that this draft yet has pretensions to be a standard, but rather indicates what will become the case if and when it is accepted as an RFC with the status of a proposed or draft standard.]

[Remarks enclosed in square brackets and aligned with the left margin, such as this one, are not part of this draft, but are editorial notes to explain matters amongst ourselves, or to point out alternatives, or to assist the RFC Editor.]

[In this draft, references to [NNTP] are to be replaced by [RFC 977], or else by references to the RFC arising from the series of drafts draft-ietf-nntpext-base-*.txt, in the event that such RFC has been accepted at the time this document is published.
References to [KLYNE], which has now passed its IETF last call, are to be replaced by references to the eventual RFC.]

Table of Contents

1. Introduction

1.1. Basic Concepts

"Netnews" is a set of protocols for generating, storing and retrieving news "articles" (which resemble email messages) and for exchanging them amongst a readership which is potentially widely distributed. It is organized around "newsgroups", with the expectation that each reader will be able to see all articles posted to each newsgroup in which he participates. These protocols most commonly use a flooding algorithm which propagates copies throughout a network of participating servers. Typically, only one copy is stored per server, and each server makes it available on demand to readers able to access that server.

An important characteristic of Netnews is the lack of any requirement for a central administration or for the establishment of any controlling host to manage the network. A network which limits participation to some restricted set of hosts (within some company, for example) is a "closed" network; otherwise it is an "open" network. A set of hosts within a network which, by mutual arrangement, operates some variant (whether more or less restrictive) of the Netnews protocols is a "cooperating subnet".

"Usenet" is a particular worldwide open network based upon the Netnews protocols, with the newsgroups being organized into recognized "hierarchies". Anybody can join (it is simply necessary to negotiate an exchange of articles with one or more other participating hosts).

A "policy" is a rule intended to facilitate the smooth operation of a network by establishing parameters which restrict behaviour that, whilst technically unexceptionable, would nevertheless contravene some accepted standard of "Good Netkeeping". Since the ultimate beneficiaries of a network are its human readers, who will be less tolerant of poorly designed interfaces than mere computers, articles in breach of established policy can cause considerable annoyance to their recipients.

1.2. Objectives

The purpose of this present standard is to define the format of articles and the protocols to be used for Netnews in general, and for Usenet in particular, and to set standards to be followed by software that implements those protocols.

It is NOT the purpose of this standard to settle matters of policy, nor aspects of software behaviour which do not impinge upon the generation, transmission, storate and reception of articles, nor how

the authority of various agencies to exercise control or oversight of the various parts of Usenet is established. For these purposes, a separate Best Current Practice document [USEAGE] is being provided.

Nevertheless, it is assumed that agencies with the necessary authority will exist, and tools are provided within the protocols for their use.

1.3. Historical Outline

Network news originated as the medium of communication for Usenet, circa 1980. Since then, Usenet has grown explosively, and many Internet and non-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.1 and Appendix A.2 for historical reasons only. With publication of [RFC 850] 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" [Son-of-1036] was written in 1994 by Henry Spencer. That document formed the original basis for this standard, and its author has endorsed this standard as its successor. Much is taken directly from Son of 1036, and it is hoped that we have followed its spirit and intentions. It is anticipated that [Son-of-1036] will be published as an Historic RFC, in a suitably relabelled form, following the publication of this standard.

1.4. Transport

As in this standard's predecessors, the exact means used to transmit articles from one host to another is not specified. NNTP [NNTP] is the most common transmission method on the Internet, but much transmission takes place entirely independent of the Internet. Other methods in use include the UUCP protocol [RFC 976] extensively used in the early days of Usenet, FTP, downloading via satellite, tape archives, and physically delivered magnetic and optical media.

2. Definitions, Notations and Conventions

2.1. Definitions

An "article" is the unit of news, analogous to an [RFC 2822] "message". A "proto-article" is one that has not yet been injected into the news system.

A "message identifier" (5.3) is a unique identifier for an article, usually supplied by the "posting agent" which posted it or, failing

that, by the "injecting agent". It distinguishes the article from every other article ever posted anywhere. Articles with the same message identifier are treated as if they are the same article regardless of any differences in the body or headers.

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 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 "hierarchy" is the set of all newsgroups whose names share a first component (as defined in 5.5). The term "sub-hierarchy" is also used where several initial components are shared.

A "poster" is the person or software that composes and submits a possibly compliant article to a "posting agent". The poster is analogous to [RFC 2822]'s author.

A "posting agent" is the software that assists posters to prepare proto-articles, in compliance with this standard. The proto-article is then passed on to an "injecting agent" for final checking and injection into the news stream. If the article is not compliant, or is rejected by the injecting agent, then the posting agent informs the poster with an explanation of the error.

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

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

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.

An (email) "address" is the mailbox [RFC 2822] (or more particularly the addr-spec within that mailbox) which directs the delivery of an email to its intended recipient, who is said to "own" that address.

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 (5.2), unless it also has a Reply-To-header (6.1).

A "sender" is the person or software (usually, but not always, the same as the poster) responsible for the operation of the posting agent or, which amounts to the same thing, for passing the article to the injecting agent. The sender is analogous to [RFC 2822]'s sender.

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 "news database" is the set of articles and related structural information stored by a serving agent and made available for access by reading agents.

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

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 standard's news format and those using other formats.

2.2. Textual Notations

This standard 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.

"US-ASCII" is short for "the ANSI X3.4 character set" [ANSI X3.4]. While "ASCII" is often misused to refer to various character sets somewhat similar to X3.4, in this standard "US-ASCII" is used to mean X3.4 and only X3.4. US-ASCII is a 7 bit character set. Please note that this standard 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, when capitalized, are used to define the significance of individual requirements. The key words "MUST", "REQUIRED", "SHOULD", "RECOMMENDED", "MAY" and "OPTIONAL", and any of those words

associated with the word "NOT", are to be interpreted as described in [RFC 2119].

> NOTE: The use of "MUST" or "SHOULD" implies a requirement that would or could lead to interoperability problems if not followed.
>
> NOTE: A requirement imposed on a relaying or serving agent regarding some particular article should be understood as applying only if that article is actually accepted for processing (since any agent may always reject any article entirely, for reasons of site policy).

Wherever the context permits, use of the masculine includes the feminine and use of the singular includes the plural, and vice versa.

Throughout this standard we will give examples of various definitions, headers and other specifications. It needs to 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 ".example" is used in all sample domains and addresses. The hierarchy "example.*" is also used as a sample hierarchy. Information on the ".example" top level domain is in [RFC 2606].

2.3. Relation To Email and MIME

The primary intent of this standard is to describe the news article format. Insofar as news articles are a subset of the email message format augmented by some new headers, this standard incorporates many (though not all) of the provisions of [RFC 2822], with the aim of enabling news articles to pass through email systems and vice versa, provided only that they contain the minimum headers required for the mode of transport being used. Unfortunately, the match is not perfect, but it is the intention of this standard that gateways between Email and Netnews should be able to operate with the minimum of tinkering.

Likewise, this standard incorporates (see section 6.21) many, though not all, of the provisions of the MIME standards [RFC 2045] et seq which, though designed with Email in mind, are mostly applicable to Netnews.

2.4. Syntax

The complete syntax defined in this standard is repeated, for convenience, in Appendix B.

2.4.1. Syntax Notation

This standard uses the Augmented Backus Naur Form described in [RFC 2234].

In particular, it makes significant use of the "incremental alternative" feature of that notation. For example, the two rules

```
header              = other-header
header              =/ Date-header
```

are equivalent to the single rule

```
header              = other-header / Date-header
```

2.4.2. Syntax copied from other standards

The following syntactic forms for characters, atoms and folding, taken from [RFC 2234] or from [RFC 2822] and adapted to incorporate variations introduced in [RFC 2047], 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,
                    %d14-31 /          ; and whitespace characters
                    %d127
   specials       = "(" / ")" /        ; Special characters used in
                    "<" / ">" /        ; other parts of the syntax
                    "[" / "]" /
                    ":" / ";" /
                    "@" / "\" /
                    "," / "." /
                    DQUOTE
   WSP            = SP / HTAB          ; whitespace characters
   FWS            = ([*WSP CRLF] 1*WSP); folding whitespace
   ccontent       = ctext / quoted-pair / comment / encoded-word
   comment        = "(" *([FWS] ccontent) [FWS] ")"
   CFWS           = *([FWS] comment) ( ([FWS] comment) / FWS )
   DQUOTE         = %d34               ; quote mark
   quoted-pair    = "\" text
   atext          = ALPHA / DIGIT /
                    "!" / "#" /        ; Any US-ASCII character except
                    "$" / "%" /        ; controls, SP, and specials.
                    "&" / "'" /        ; Used for atoms
                    "*" / "+" /
                    "-" / "/" /
                    "=" / "?" /
                    "^" / "_" /
```

```
                    "`" / "{" /
                    "|" / "}" /
                    "~"
      atom          = [CFWS] 1*atext [CFWS]
      dot-atom      = [CFWS] dot-atom-text [CFWS]
```

```
dot-atom-text = 1*atext *( "." 1*atext )
qcontent      = qtext / quoted-pair
quoted-string = [CFWS] DQUOTE
                   *( [FWS] qcontent ) [FWS]
                   DQUOTE [CFWS]
word          = atom / quoted-string
```

NOTE: Following [RFC 2234], literal text included in the syntax is to be regarded as case-insensitive. However, in contradistinction to [RFC 2822], the Netnews protocols are sensitive to case in some instances (as in newsgroup-names, some header parameters, etc.). Care has been taken to indicate this explicitly where required.

As in [RFC 2822], where any quoted-pair appears it is to be interpreted as its text character alone. That is to say, the "\" character that appears as part of a quoted-pair is semantically "invisible".

Again, as in [RFC 2822], strings of characters that include characters not syntactically allowed in some particular context may be incorporated into a quoted-string by "encapsulating" them between quote (DQUOTE, US-ASCII 34) characters, prefixing every quote and backslash character (and possibly other characters too) with a "\" so as to form a quoted-pair, and possibly introducing folding by prefixing some WSP with CRLF.

The semantic value of a quoted-string (i.e. the result of reversing the encapsulation) is a string of characters which includes neither the optional CFWS outside of the quote characters, nor the quote characters themselves, nor any CRLF contained within any FWS between the two quote characters, nor the "\" which introduces any quoted-pair.

The following basic forms are taken from [RFC 2045] and [RFC 2231] (having regard to [RFC Errata]), adapted to use the folding syntax from [RFC 2822]:

```
charset       = <registered character set name>
                                    ;  [RFC 2978]
language      = <registered language tag>
                                    ;  [RFC 3066]
encoded-word  = "=?" charset ["*" language] "?" encoding
                    "?" encoded-text "?="
parameter     = regular-parameter / extended-parameter
regular-parameter       = [CFWS] regular-parameter-name [CFWS]
                    "=" value
regular-parameter-name  = attribute [section]
attribute     = 1*attribute-char
```

```
attribute-char= <any (US-ASCII) CHAR except SPACE, CTLs,
                   "*", "'", "%", or tspecials>
tspecials     = "(" / ")" / "<" / ">" / "@" /
                "," / ";" / ":" / "\" / DQUOTE /
                "/" / "[" / "]" / "?" / "="
```

```
extended-parameter
              = ( [CFWS] extended-initial-name [CFWS]
                   "=" extended-initial-value ) /
                ( [CFWS] extended-other-names [CFWS]
                   "=" extended-other-values )
value         = [CFWS] token [CFWS] / quoted-string
token         = 1*<any (US-ASCII) CHAR except SP, CTLs,
                   or tspecials>
```

For the purposes of section 5 of [RFC 2047] all headers (4.1) defined in this standard are to be considered as "extension message header fields" (insofar as they are not already so considered under the existing Email standards), permitting the use of [RFC 2047] encodings within any unstructured header, or within any comment or phrase permitted within any structured header.

The syntax of encoded-text and of encoding can be found in [RFC 2047], and there are restrictions on the characters that may occur within an encoded-text, depending on its context. There are also restrictions on the overall length of an encoded-word and of headers containing encoded-words and requirements for encoded-words to have FWS on either side of them in most contexts. All these restrictions and requirements MUST be observed.

2.4.3. Syntax adapted from Email and MIME

Much of the syntax of Netnews Articles is based on the corresponding syntax defined in [RFC 2822]. Therefore, wherever in this standard the syntax is stated to be taken from [RFC 2822], it is to be understood, unless explicitly stated to the contrary, as the syntax defined by [RFC 2822], but NOT including any syntax defined in section 4 ("Obsolete syntax") of [RFC 2822]. 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.

Likewise, certain syntax from the MIME specifications [RFC 2045] et seq is also considered to have been incorporated into this standard (see 6.21).

However, there are differences arising from some special requirements of Netnews, and the following syntactic rules therefore supersede the corresponding rules given in [RFC 2822].

> NOTE: Netnews parsers historically have been much less permissive than Email parsers, and this is reflected in the modifications referred to, and in some further specific rules.

In contradistinction to [RFC 2822], an unstructured header (e.g. a Subject-header) MUST contain at least one non-whitespace character.

```
unstructured    = 1*( [FWS] ( utext / encoded-word ) ) [FWS]
```

Extended-phrases (known somewhat confusingly in [RFC 2822] as obs-phrases) are introduced to allow headers such as

```
   From: Joe Q. Public <joe@public.example>
```

without the necessity of using a quoted-string. They MUST be accepted by compliant software, but they SHOULD NOT be generated until software capable of accepting them has become widely deployed.

```
   phrase          = 1*( [CFWS] encoded-word [CFWS] / word ) /
                     extended-phrase
   extended-phrase = ( [CFWS] encoded-word [CFWS] / word )
                        *( [CFWS] encoded-word [CFWS] / word  /
                           [CFWS] "." [CFWS] )
```

Within a date-time, two of the obs-zones from [RFC 2822] are retained because of current widespread usage.

```
   zone            = (( "+" / "-" ) 4DIGIT) / "UT" / "GMT"
```

The forms "UT" and "GMT" (indicating universal time) are to be regarded as obsolete synonyms for "+0000". They MUST be accepted, and passed on unchanged, by all agents, but they MUST NOT be generated as part of new articles by posting and injecting agents.

Msg-ids are redefined to be a "normalized" subset of those defined by [RFC 2822], ensuring that no string of characters is quoted unless strictly necessary (it must contain at least one mqspecial) and no single character is prefixed by a "\" in the form of a quoted-pair unless strictly necessary, and moreover there is no possibility for ">" or WSP to occur inside a msg-id, whether quoted or not. Thus, whereas under [RFC 2822]

```
   <abcd@example.com>
   <"abcd"@example.com>
   <"ab\cd"@example.com>
```

would be considered semantically equivalent, only the first of them is syntactically permitted by this standard, and hence a simple comparison of octets will always suffice to determine the identity of two msg-ids.

```
   msg-id          = "<" id-left "@" id-right ">"
   id-left         = dot-atom-text / no-fold-quote
   id-right        = dot-atom-text / no-fold-literal
   no-fold-quote   = DQUOTE
                        *( mqtext / "\\" / "\" DQUOTE )
                        mqspecial
                        *( mqtext / "\\" / "\" DQUOTE )
                        DQUOTE
   mqtext          = NO-WS-CTL /      ; all of <text> except
                     %d33 /           ; SP, HTAB, "\", ">"
                     %d35-61 /        ; and DQUOTE
```

%d63-91 /
%d93-126

```
mqspecial       = "(" / ")" /      ; same as specials except
                  "<" /            ; "\" and DQUOTE quoted
                  "[" / "]" /      ; and ">" omitted
                  ":" / ";" /
                  "@" / "\\" /
                  "," / "." /
                  "\" DQUOTE
no-fold-literal = "[" *( mdtext / "\[" / "\]" / "\\" ) "]"
mdtext          = NO-WS-CTL /      ; Non white space controls
                  %d33-61 /        ; The rest of the US-ASCII
                  %d63-90 /        ; characters not including
                  %d94-126         ; ">", "[", "]", or "\"
```

2.5. Language

Various constant strings in this standard, 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 as transmitted MUST be the English-derived ones defined in this standard.

3. Changes to the existing protocols

This standard prescribes many changes, clarifications and new features since the protocols described in [RFC 1036] and [Son-of-1036]. It is the intention that they can be assimilated into Usenet as it presently operates without major interruption to the service, though some of the new features may not begin to show benefit until they become widely implemented. This section summarizes the main changes, and comments on some features of the transition.

3.1. Principal Changes

- The [RFC 2822] conventions for parenthesis-enclosed comments in headers are supported.
- Whitespace is permitted in Newsgroups-headers, permitting folding of such headers. Indeed, all headers can now be folded.
- An enhanced syntax for the Path-header enables the injection point of and the route taken by an article to be determined with certainty.
- Large parts of MIME are recognized as an integral part of Netnews.
- There is a new Control message 'mvgroup' to facilitate moving a group to a different place (name) in a hierarchy.
- There is a new mandatory Injection-Date-header to facilitate the rejection of stale articles.

- There are several new optional headers defined, notably Archive, Complaints-To, Injection-Info, Mail-Copies-To, Posted-And-Mailed and User-Agent, leading to increased functionality.
- Provision has been made for almost all headers to have MIME-style parameters (to be ignored if not recognized), thus facilitating

extension of those headers in future standards.
- o Certain headers and Control messages (Appendix A.3 and Appendix A.4) have been made obsolete.
- o Distributions are expected to be checked at the receiving end, as well as the sending end, of a relaying link.
- o There are numerous other small changes, clarifications and enhancements.

3.2. Transitional Arrangements

An important distinction must be made between serving and relaying agents, which are responsible for the distribution and storage of news articles, and user agents, which are responsible for interactions with users. It is important that the former should be upgraded to conform to this standard as soon as possible to provide the benefit of the enhanced facilities. Fortunately, the number of distinct implementations of such agents is rather small, at least so far as the main "backbone" of Usenet is concerned, and many of the new features are already supported. Contrariwise, there are a great number of implementations of user agents, installed on a vastly greater number of small sites. Therefore, the new functionality has been designed so that existing agents may continue to be used, although the full benefits may not be realised until a substantial proportion of them have been upgraded.

In the list which follows, care has been taken to distinguish the implications for both kinds of agent.

- o [RFC 2822] style comments in headers do not affect serving and relaying agents (note that the Message-ID-, Newsgroups-, Distribution- and Path-headers do not contain them). They are unlikely to hinder their proper display in existing reading agents except in the case of the References-header in agents which thread articles. Therefore, it is provided that they SHOULD NOT be generated except where permitted by the previous standards.
- o Because of its importance to all serving agents, the extension permitting whitespace and folding in Newsgroups-headers SHOULD NOT be used until it has been widely deployed amongst relaying agents. User agents are unaffected.
- o The new style of Path-header is already consistent with the previous standards. However, the intention is that relaying agents should eventually reject articles in the old style, and so this possibility should be offered as a configurable option in relaying agents. User agents are unaffected.
- o The introduction of MIME reflects a practice that is already widespread. Articles in strict compliance with the previous standards (using strict US-ASCII) will be unaffected. Many user agents already support it, at least to the extent of widely used

charsets such as ISO-8859-1. Users expecting to read articles using other charsets will need to acquire suitable reading agents. It is not intended, in general, that any single user agent will be able to display every charset known to IANA, but all such agents MUST support US-ASCII. Serving and relaying

agents are not affected.

- The new Control: mvgroup command will need to be implemented in serving agents. For the benefit of older serving agents it is therefore RECOMMENDED that it be followed shortly by a corresponding newgroup command and it MUST always be followed by a rmgroup command for the old group after a reasonable overlap period. An implementation of the mvgroup command as an alias for the newgroup command would thus be minimally conforming. User agents are unaffected.
- Provision is made for relaying and serving agents to use the Date-header in the case of articles injected through existing agents which do not provide an Injection-Date-header.
- All the headers newly introduced by this standard can safely be ignored by existing software, albeit with loss of the new functionality.

4. Basic Format

4.1. Syntax of News Articles

The overall syntax of a news article is:

```
article           = 1*( header CRLF ) separator body
header            = other-header
other-header      = header-name ":" 1*SP other-content
header-name       = 1*name-character *( "-" 1*name-character )
name-character    = ALPHA / DIGIT
other-content     = <the content of a header defined by some
                     other standard>
separator         = CRLF
body              = *( *998text CRLF )
```

However, the rule given above for header is incomplete. Further alternatives will be added incrementally as the various Netnews headers are introduced in this standard (or in future extensions), using the "=/" notation defined in [RFC 2234]. For example, a putative "Usenet-header" would be defined as follows:

```
header            =/ Usenet-header
Usenet-header     = "Usenet" ":" SP Usenet-content
                        *( ";" ( Usenet-parameter /
                                 extension-parameter ) )
Usenet-content    = <syntax specific to that Usenet-header>
Usenet-parameter  = <a parameter specific to that Usenet-header>
```

where the Usenet-parameter, which MUST always be of the same syntactic form as a parameter, is only provided for certain headers, and even the extension-parameter is omitted in some cases (see 4.2.2). Observe that "Usenet" is (and MUST be) of the syntactic form

of a header-name.

```
   extension-parameter
                     = <a parameter not defined by this standard>
   x-attribute     = "x-" attribute
```

An article consists of some headers followed by a body. An empty line separates the two. A header begins with a header-name identifying it, and can be continued onto subsequent lines as described in section 4.2.3. 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 [RFC 2822] 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.

> NOTE: The syntax above defines the canonical form of a news article as a sequence of lines each terminated by CRLF. This does not prevent serving or relaying agents from storing or handling the article in other formats (e.g. using a single LF in place of CRLF) so long as the overall effects achieved are as defined by this standard when operating on the canonical form.

4.2. Headers

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

4.2.1. Naming of Headers

Despite the restrictions on header-name syntax imposed by the grammar, relaying, serving and reading agents SHOULD tolerate header-names containing any US-ASCII printable character other than colon (":", US-ASCII 58).

Whilst relaying agents MUST accept, and pass on unaltered, any non-variant header whose header-name is syntactically correct, and reading agents MUST at least provide the option of displaying them, posting and injecting agents SHOULD NOT generate headers other than

- headers established by this standard or any extension to it;
- those recognized by other IETF-established standards, notably the Email standard [RFC 2822] and its extensions, and included in the Permanent Message Header Field Repository maintained by IANA in accordance with [KLYNE], but excluding any header explicitly deprecated for Netnews (e.g. see section 9.2.1 for the deprecated Disposition-Notification-To-header);
- experimental headers beginning with "X-" (as defined in 4.2.5.1);

- on a provisional basis only, headers related to new protocols under development which are listed in the Provisional Message Header Field Repository maintained by IANA in accordance with [KLYNE].

However, software SHOULD NOT attempt to interpret headers not

specifically intended to be meaningful in the Netnews environment.

Header-names are case-insensitive. There is a preferred case convention set out in [USEAGE], and which is used in the various rules defining headers in this standard. Relaying and reading agents MUST, however, tolerate header-names in any case.

4.2.2. MIME-style Parameters

A few header-specific MIME-style parameters are defined in this standard (see 6.12 and 6.19), but there is also provision for generic extension-parameters to appear in most headers for the purpose of allowing future extensions to those headers. Observe that such parameters do not, in general, occur in headers defined in other standards, except for the MIME standards [RFC 2045] et seq. and their extensions.

Extension-parameters, other than those using x-attributes, MUST NOT be used unless they have first been defined in an IETF-approved RFC (whether Informational, Experimental or Standards-Track) or, on a provisional basis only, in relation to new protocols under development which are the subject of (or intended to be the subject of) some such IETF-approved RFC. They MUST ONLY be defined for use in those headers where the syntax of this standard so allows. They SHOULD NOT, at present, be defined for use in headers in widespread use prior to the introduction of this standard (this restriction is likely to be removed in a future version of this standard). Nevertheless, compliant software MUST accept such parameters wherever syntactically allowed in this standard (ignoring them if their meaning is unknown) and SHOULD accept (and ignore) them in all structured headers wherever defined.

[We could go further, and establish an IANA registry for these parameters, preloaded with the ones already defined in this standard. A good model for setting up such a registry is to be found in RFC 2183 (Content-Disposition).]

> NOTE: The syntax does not permit extension-parameters in unstructured headers (where they are unnecessary) or in certain headers (notably the Date-, From-, Message-ID-, Reply-To-, Sender-, Keywords-, Mail-Copies-To-, References-, Supersedes- and Complaints-To-headers) which are the same (or similar to) headers already existing in the Email standards.

Each header-specific MIME-style parameter introduced in this standard is described by specifying

(a) its attribute, and
(b) the syntax rule(s) defining the object(s) permitted in its value.

Any parameter, or set of parameters with numbered sections, which,

according to [RFC 2231], is semantically equivalent to an unnumbered regular-parameter with that attribute and value may be used.

NOTE: If the value is not of the syntactic form of a token and is not encoded as an extended-value, it is necessary to encapsulate it within a quoted-string (2.4.2). Observe that the syntax of a parameter also allows additional WSP, folding and comments.

The semantics of a parameter is always to associate its attribute with the object represented by the token, or the semantic value (2.4.2) of the quoted-string, contained in its value.

For example, the posting-sender-parameter (6.19) is defined to be
<a parameter with attribute "sender" and value some sender-value>
where
sender-value = mailbox / "verified"
A valid posting-sender-parameter would be
sender = "\"Joe D. Bloggs\" <jdbloggs@bloggs.example>" (authinfo)
The comment (syntactically part of the quoted-string) is irrelevant. The actual mailbox (to be used, for example, if email is to be sent to the sender) is
"Joe D. Bloggs" <jdbloggs@bloggs.example>

4.2.3. White Space and Continuations

Each header is logically a single line of characters comprising the header-name, the colon with its following SP, the content, and any parameters. For convenience, however, the content and parameters can be "folded" into a multiple line representation by inserting a CRLF before any WSP contained within any FWS (or CFWS), but not any other SP or HTAB, allowed by this standard (see 2.4.2). For example, the header:
Approved: modname@modsite.example (Moderator of example.foo.bar)
can be represented as:
Approved: modname@modsite.example
(Moderator of example.foo.bar)

FWS occurs at many places in the syntax (usually within a CFWS) in order to allow the inclusion of comments, whitespace and folding. The syntax is in fact ambiguous insofar as it sometimes allows for two consecutive FWS (as least one of which is always optional), or for an optional FWS followed by an explicit CRLF. In the first case, (one of) the optional FWS MUST NOT be instantiated; in the second case, the [*WSP CRLF] within the optional FWS MUST NOT be instantiated. Thus it is thus precluded that any line of a header should be made up of whitespace characters and nothing else (for such a line might otherwise have been interpreted by a non-compliant agent as the separator between the headers and the body of the article).

NOTE: This does not lead to semantic ambiguity because, unless specifically stated otherwise, the presence or absence of

folding, a comment or additional WSP has no semantic meaning and, in particular, it is a matter of indifference whether it forms a part of the syntactic construct preceding it or the one following it.

> NOTE: It may be observed that the content part of every header begins and ends with an optional CFWS (or FWS in the case of a few headers). Moreover, every parameter also begins and ends with an optional CFWS.

In accordance with the syntax, the header-name and colon on the first line MUST be followed by a SP (even if the rest of the header is empty, which in fact cannot occur because this standard defines no headers with empty content and extensions MUST NOT do so). Even though the syntax allows otherwise, at least some visible content MUST appear on that first line (to avoid the possibility of harm by any non-compliant agent that might eliminate a trailing WSP). Posting and injecting agents are REQUIRED to enforce these restrictions, deleting any headers with empty content that are encountered, but relaying and serving agents MUST accept and pass on untouched articles that violate them.

> NOTE: This standard differs from [RFC 2822] in requiring that SP following the colon (it was also an [RFC 1036] requirement).

Posters and posting agents SHOULD use SP, not HTAB, where white space is desired in headers (some existing software expects this). Relaying and serving agents SHOULD accept HTAB in all such cases, however.

Relaying and serving agents MUST NOT refold headers (i.e. they must pass on the folding as received).

4.2.4. Comments

Strings of characters which are treated as comments may be included in headers wherever the syntactic element CFWS occurs. They consist of characters enclosed in parentheses. Comments may be nested.

> NOTE: Although CFWS occurs wherever whitespace is allowed in almost all headers, there are exceptions where only FWS is permitted (hence folding but no comments). Notably, this happens in the case of the Message-ID-, Newsgroups-, Distribution-, Path- and Followup-To-headers, and within the Date-header except right at the end.

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 content of the comment; the content 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, namely following mailboxes in From and similar headers (a now deprecated convention for indicating the owner of that mailbox), and to indicate the name of the timezone in Date-headers. However, compliant software MUST accept them in all

places where they are syntactically allowed.

4.2.5. Header Properties

There are three special properties that may apply to particular headers, namely: "experimental", "inheritable", and "variant". When a header is defined, in this (or any future) standard, as having one (or possibly more) of these properties, it is subject to special treatment, as indicated below.

4.2.5.1. Experimental Headers

Experimental headers are those whose header-names begin with "X-". They are to be used for experimental Netnews features, or for enabling additional material to be propagated with an article. They are not (and will not be) defined by this, or any, standard.

> NOTE: Experimental headers are suitable for situations where they need only to be human readable. They are not intended to be recognized by widely deployed Netnews software and, should such a requirement be envisaged, it is preferable to arrange for a suitable normal header to be included in the Provisional Message Header Field Repository maintained by IANA in accordance with [KLYNE].

4.2.5.2. Inheritable Headers

Subject only to the overriding ability of the poster to determine the contents of the headers in a proto-article, headers with the inheritable property MUST be copied by followup agents (perhaps with some modification) into the followup article, and headers without that property MUST NOT be so copied. Examples include:

- Newsgroups (5.5) - copied from the precursor, subject to any Followup-To-header.
- Subject (5.4) - often modified by prefixing with "Re: ", but otherwise copied from the precursor.
- References (6.10) - copied from the precursor, with the addition of the precursor's Message-ID.
- Distribution (6.6) - copied from the precursor.

> NOTE: The Keywords-header is not inheritable, though some older newsreaders treated it as such.

4.2.5.3. Variant Headers

Headers with the variant property may differ between (or even be completely absent from) copies of the same article as stored or relayed throughout a Netnews system. The manner of the difference (or absence) MUST be as specified in this (or some future) standard. Typically, these headers are modified as articles are propagated, or

they reflect the status of the article on a particular serving agent, or cooperating group of such agents. The variant header MAY be placed anywhere within the headers (though placing it first is recommended). The principal examples are:

- Path (5.6) - augmented at each relaying agent that an article passes through.
- Xref (6.16) - used to keep track of the article locators of crossposted articles so that reading agents serviced by a particular serving agent can mark such articles as read.

4.3. Body

The body of an article SHOULD NOT be empty. A posting or injecting agent which does not reject such an article entirely SHOULD at least 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 in this situation.

Note that an article body is a sequence of lines terminated by CRLFs, not arbitrary binary data, although a MIME Content-Type-header may impose some structure or intended interpretation upon it. In particular the body MUST end with a CRLF.

4.4. Octets, Characters and Character Sets

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

> NOTE: this corresponds to the range of octets permitted for MIME "8bit data" [RFC 2045]. Thus raw binary data cannot be transmitted in an article body except by the use of a Content-Transfer-Encoding such as base64.

In particular, transmission paths MUST convey all headers (including body part headers and headers within message/rfc822 objects) intact, even if they contain octets in the range 128 to 255. These requirements include the transmissiom paths between posting agents, injecting agents, relaying agents, serving agents and reading agents, but NOT the paths traversed by Netnews articles that have been gatewayed into Email (8.8.1).
[At some point it will be necessary for the IMAP standards to catch up with these requirements.]

Moreover, a body or body part that uses Content-Transfer-Encoding 8bit MUST NOT have that encoding changed to quoted-printable or bas64 during transmission, except in the course of gatewaying into some

other medium such as email (8.8).

NOTE: An agent that was incapable of handling 8bit would be unlikely to be able to make use of the article on its own servers, and the usual flooding algorithm would likely find some alternative route to get the article to destinations where it is needed.

4.4.1. Character Sets within Article Headers

The character set for headers is US-ASCII. Where the use of non-ASCII characters is required, they MUST be encoded using the MIME mechanisms defined in [RFC 2047] and [RFC 2231].

Examples:

```
Organization: Technische =?iso-8859-1?Q?Universit=E4t_M=FCnchen?=
Approved: =?iso-8859-1?Q?Fran=E7ois_Faur=E9?= <ff@modsite.example>
   (=?iso-8859-1?Q*fr?Mod=E9rateur_autoris=E9?=)
Archive: yes; filename*=iso-8859-1'es'ma=F1ana.txt
```

NOTE: The raw use of non-ASCII character sets or of encodings other than those described above is not compliant with this standard, even though such usage has been seen in some hierarchies (with no indication of which character set has been used beyond the user's ability to guess based upon other clues in the article, or custom within the newsgroup). Future extensions to this standard may make provision for other character sets, hence the requirement that octets beyond the US-ASCII range be transported without error.

4.4.2. Character Sets within Article Bodies

Within article bodies, characters are represented as octets according to the encoding scheme implied by any Content-Transfer-Encoding- and Content-Type-headers [RFC 2045]. In the absence of such headers, reading agents cannot be relied upon to display correctly more than the US-ASCII characters, though they MUST display at least those.

NOTE: The use of non-ASCII characters in the absence of an appropriate Content-Type-header is not compliant with this standard, even though such usage has been seen in some hierarchies (with no indication of which character set has been used beyond the user's ability to guess based upon other clues in the article, or custom within the newsgroup).

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.

4.5. Size Limits

Compliant software MUST support headers of at least 998 octets, and

that is the only limit on the length of a header line prescribed by this standard. However, specific rules to the contrary may apply in particular cases (for example, according to [RFC 2047] header lines containing encoded-words are limited to 76 octets).

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).

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 body lines of at least that length, 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 [RFC 2822]. In practice, lines will be much shorter, and [USEAGE] suggests a default limit of 79 characters to be used where there are no pressing needs to do otherwise.

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.

4.6. 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@site1.example>
From: Ann Example <a.example@site1.example>
Subject: Announcing a new sample article.
Date: Wed, 27 Mar 2002 12:12:50 +0300
Approved: example.announce moderator <jsmith@site.example>
Followup-To: example.chat
Reply-To: Ann Example <a.example+replies@site1.example>
Expires: Mon, 22 Apr 2002 12:12:50 +0300
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.
Injection-Info: injector.site.example; posting-host=du003.site.example
Complaints-To: abuse@site.example
Injection-Date: Mon, 22 Apr 2002 13:22:40 +0300
```

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

Ann.

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

[The RFC Editor is invited to change the above Date, Injection-Date and Expires headers to match the actual publication dates and to insert its correct URL.]

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 also that there are situations, discussed in the relevant parts of section 6, where References-, Sender-, or Approved-headers are mandatory.

A proto-article (8.2.1) may lack some of these mandatory headers, but they MUST then be supplied by the injecting agent.

5.1. Date

The Date-header contains the date and time that the article was prepared by the poster ready for transmission and SHOULD express the poster's local time. The content syntax makes use of syntax defined in [RFC 2822] (but see the revised definition of zone in section 2.4.3).

```
header              =/ Date-header
Date-header         = "Date" ":" SP Date-content
Date-content        = date-time
```

The date-time MUST be semantically valid as required by [RFC 2822]. Although folding white space is permitted throughout the date-time syntax, it is RECOMMENDED that a single space be used in each place that FWS appears (whether it is required or optional).

The date-time MUST NOT be more than 24 hours into the future (injecting agents will reject such articles, possibly even when the margin is less than 24 hours - see 8.2.2 - and MAY also reject them if it is inordinately far into the past). Relaying agents MUST NOT modify the Date-header in transit.

5.1.1. Examples

```
Date: Sat, 26 May 2001 11:13:00 -0500 (EST)
Date: 26 May 2001 16:13 +0000
Date: 26 May 2001 16:13 GMT (Obsolete)
```

5.2. From

The From-header contains the email address(es), possibly including the full name(s), of the article's poster(s), or of the person or agent on whose behalf the article is posted (see the Sender-header, 6.2). The content syntax makes use of syntax defined in [RFC 2822].

```
header              =/ From-header
From-header         = "From" ":" SP From-content
From-content        = mailbox-list
```

NOTE: Observe that there is no provision for parameters in this header, or in other headers containing addresses likely to be used for sending email (see 4.2.2). When, for whatever reason, a poster does not wish to use a valid address, the mailbox concerned SHOULD end in the top level domain ".invalid" [RFC 2606].

5.2.1. Examples:

```
From: John Smith <jsmith@site.example>
From: "John Smith" <jsmith@site.example>, dave@isp.example
From: "John D. Smith" <jsmith@site.example>, andrew@isp.example,
   fred@site2.example
From: Jan Jones <jan@please_setup_your_system_correctly.invalid>
From: Jan Jones <joe@guess-where.invalid>
From: dave@isp.example (Dave Smith)
```

NOTE: the last example shows a now deprecated convention of putting a poster's full name in a comment following the mailbox, rather than in a phrase at the start of it. Observe also the use of the quoted-string "John D. Smith" which SHOULD still be used on account of presence of the '.' character, even though software is now REQUIRED to operate without it (see 2.4.3).

5.3. Message-ID

The Message-ID-header contains the article's message identifier, a unique identifier distinguishing the article from every other article. The content syntax makes use of syntax defined in [RFC 2822] (but see the revised definition of msg-id in section 2.4.3).

```
header              =/ Message-ID-header
Message-ID-header   = "Message-ID" ":" SP Message-ID-content
Message-ID-content = [FWS] msg-id [FWS]
```

The msg-id MUST NOT be more than 250 octets in length.

NOTE: The length restriction ensures that systems which accept message identifiers as a parameter when retrieving an article

(e.g. [NNTP]) can rely on a bounded length. Observe that msg-id includes the '<' and '>'.

> Observe that, in contrast to the corresponding header in [RFC 2822], the syntax does not allow comments within the Message-ID-header; this is to simplify processing by relaying and serving agents and to ensure interoperability with existing implementations.

An agent generating an article's message identifier MUST ensure that it is unique (as also required in [RFC 2822]) and that it is chosen in such a way that it will NEVER be applied to any other Netnews article or Email message. However, an article emailed (without encapsulation) to a moderator (8.2.2 and 8.7) or gatewayed into some other medium (8.8.1) SHOULD retain the same message identifier throughout its travels so long as it remains recognizably the same article.

Even though commonly derived from the domain name of the originating site (and domain names are case-insensitive), a message identifier MUST NOT be altered in any way during transport, or when copied (as into a References-header), and thus a simple (case-sensitive) comparison of octets will always suffice to recognize that same message identifier wherever it subsequently reappears.

> NOTE: These requirements are to be contrasted with those of the un-normalized msg-ids defined by [RFC 2822], which may perfectly legitimately become normalized (or vice versa) during transport or copying in email systems.

> NOTE: Some old software may treat message identifiers that differ only in case within their id-right part as equivalent, and implementors of agents that generate message identifiers should be aware of this.

5.4. Subject

The Subject-header contains a short string identifying the topic of the article. This is an inheritable header (4.2.5.2), normally to be copied into the Subject-header of any followup with the possible addition of a back-reference as described in B.6.

```
header              =/ Subject-header
Subject-header      = "Subject" ":" SP Subject-content
Subject-content     = unstructured
```

> NOTE: The syntax of unstructured differs from that prescribed in [RFC 2822], so ensuring that the Subject-content is not permitted to be completely empty, or to consist of WSP only.

5.5. Newsgroups

The Newsgroups-header's content specifies the newsgroup(s) in which

the article is intended to appear. It is an inheritable header (4.2.5.2) which then becomes the default Newsgroups-header of any followup, unless a Followup-To-header is present to prescribe otherwise (see 8.6). Articles MUST NOT be passed between relaying

agents or to serving agents unless the sending agent has been configured to supply and the receiving agent to receive at least one of the newsgroup-names in the Newsgroups-header.

```
   header              =/ Newsgroups-header
   Newsgroups-header   = "Newsgroups"  ":" SP Newsgroups-content
                                *( ";" extension-parameter )
   Newsgroups-content  = [FWS] newsgroup-name
                            *( [FWS] ng-delim [FWS] newsgroup-name )
                            [FWS]
   newsgroup-name      = component *( "." component )
   component           = 1*component-grapheme
   ng-delim            = ","
   component-grapheme  = DIGIT / ALPHA / "+" / "-" / "_"
[Maybe some better word for 'grapheme'.]
```

NOTE: Observe that the syntax does not allow comments within the Newsgroups-header; this is to simplify processing by relaying and serving agents which have a requirement to process this header extremely rapidly.

Components beginning with underline ("_") are reserved for use by future versions of this standard and MUST NOT occur in newsgroup-names (whether in Newsgroups-headers or in newgroup control messages (7.2.1)). However, such names MUST be accepted.

Components beginning with "+" or "-" are reserved for use by implementations and MUST NOT occur in newsgroup-names (whether in Newsgroups-headers or in newgroup control messages). Implementors may assume that this rule will not change in any future version of this standard.

NOTE: For example, implementors may safely use leading "+" and "-" to "escape" other entities within something that looks like a newsgroup-name.

The format of newsgroup-names is ultimately determined by the policies of those administrative agencies which have the responsibility for creating new newsgroups within the various hierarchies of Usenet. There are traditional, social and technical arguments why there should be restrictions on these formats (and the force of the technical ones changes over time with developments in computers and operating systems). Therefore, such administrative agencies SHOULD establish and promulgate the restrictions they intend to apply within their own hierarchies.

NOTE: These issues are discussed more fully in [USEAGE]. The following policy restrictions represent what is considered safe and appropriate at the present time. Although purely advisory,

hierarchy administrators should consider the consequences carefully before allowing them to be exceeded. They could also be taken as the defaults in unmanaged hierarchies.

1. Uppercase letters are forbidden.

2. A component name is forbidden to consist entirely of digits.

3. A component is limited to 30 component-graphemes and a newsgroup-name to 66 component-graphemes (counting also the '.'s separating the components).

Serving and relaying agents MUST accept any syntactially correct newsgroup-name even if it would violate whatever policy restrictions may be in place. Posting and injecting agents MAY enforce them (but only with the explicit agreement of the poster).

The inclusion of folding white space within a Newsgroups-content is a newly introduced feature in this standard. It MUST be accepted by all conforming implementations (relaying agents, serving agents and reading agents). Posting agents should be aware that such postings may be rejected by overly-critical old-style relaying agents. When a sufficient number of relaying agents are in conformance, posting agents SHOULD generate such whitespace in the form of <CRLF WSP> so as to keep the length of lines in the relevant headers (notably Newsgroups and Followup-To) to a reasonable length (such as 79 characters, which is likely to be displayed satisfactorily by most current reading agents). Before such critical mass occurs, injecting agents MAY reformat such headers by removing whitespace inserted by the posting agent, but relaying agents MUST NOT do so.

Posters SHOULD use only the names of existing newsgroups in the Newsgroups-header. However, it is legitimate to cross-post to newsgroups which do not exist on the posting agent's host, provided that at least one of the newsgroups DOES exist there. 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. Serving agents MUST NOT create new newsgroups simply because an unrecognized newsgroup-name occurs in a Newsgroups-header (see 7.2.1 for the correct method of newsgroup creation).

The Newsgroups-header is intended for use in Netnews articles rather than in email messages. It MAY be used in an email message to indicate that it is a copy also posted to the listed newsgroups, in which case the inclusion of a Posted-And-Mailed header (6.9) would also be appropriate. However, it SHOULD NOT be used in an email-only reply to a Netnews article (thus the "inheritable" property of this header applies only to followups to a newsgroup, and not to followups to the poster).

5.5.1. Forbidden newsgroup-names

The following forms of newsgroup-name MUST NOT be used except for the

specific purposes indicated:

- Newsgroup-names having only one component. These are reserved for newsgroups whose propagation is restricted to a single host or local network, and for pseudo-newsgroups such as "poster" (which

has special meaning in the Followup-To-header - see section 6.7), "junk" (often used by serving agents), and "control" (likewise);
- Any newsgroup-name beginning with "control." (used as pseudo-newsgroups by many serving agents);
- Any newsgroup-name containing the component "ctl" (likewise);
- "to" or any newsgroup-name beginning with "to." (reserved for the ihave/sendme protocol described in section 7.4, and for test articles sent on an essentially point-to-point basis);
- Any newsgroup-name beginning with "example." (reserved for examples in this and other standards);
- Any newsgroup-name containing the component "all" (because this is used as a wildcard in some implementations).

A newsgroup-name SHOULD NOT appear more than once in the Newsgroups-header. The order of newsgroup-names in the Newsgroups-header is not significant, except for determining which moderator to send the article to if more than one of the groups is moderated (see 8.2).

5.6. Path

The Path-header shows the route taken by an article since its entry into the Netnews system. It is a variant header (4.2.5.3), each agent that processes an article being required to add one (or more) entries to it. This is primarily to enable relaying agents to avoid sending articles to sites already known to have them, in particular the site they came from, and additionally to permit tracing the route articles take in moving over the network, and for gathering Usenet statistics. Finally the presence of a '%' path-delimiter in the Path-header can be used to identify an article injected in conformance with this standard.

5.6.1. Format

```
header          =/ Path-header
Path-header     = "Path" ":" SP Path-content
                      *( ";" extension-parameter )
Path-content    = [FWS]
                      *( path-identity [FWS] path-delimiter [FWS] )
                      tail-entry [FWS]
path-identity   = ( ALPHA / DIGIT )
                      *( ALPHA / DIGIT / "-" / "." / ":" / "_" )
path-delimiter  = "/" / "?" / "%" / "," / "!"
tail-entry      = path-identity
```

NOTE: A Path-content will inevitably contain at least one path-identity, except possibly in the case of a proto-article that has not yet been injected onto the network.

NOTE: Observe that the syntax does not allow comments within the

Path-header; this is to simplify processing by relaying and injecting agents which have a requirement to process this header extremely rapidly.

A relaying agent SHOULD NOT pass an article to another relaying agent whose path-identity (or some known alias thereof) already appears in the Path-content. Since the comparison may be either case sensitive or case insensitive, relaying agents SHOULD NOT generate a name which differs from that of another site only in terms of case.

A relaying agent MAY decline to accept an article if its own path-identity is already present in the Path-content or if the Path-content contains some path-identity whose articles the relaying agent does not want, as a matter of local policy.

> NOTE: This last facility is sometimes used to detect and decline control messages (notably cancel messages) which have been deliberately seeded with a path-identity to be "aliased out" by sites not wishing to act upon them.

5.6.2. Adding a path-identity to the Path-header

When an injecting, relaying or serving agent receives an article, it MUST prepend its own path-identity followed by a path-delimiter to the beginning of the Path-content. In addition, it SHOULD then add CRLF and WSP if it would otherwise result in a line longer than 79 characters.

The path-identity added MUST be unique to that agent. To this end it SHOULD be one of:

1. A fully qualified domain name (FQDN) associated (by the Internet DNS service [RFC 1034]) with an A record, which SHOULD identify the actual machine prepending this path-identity. Ideally, this FQDN should also be "mailable" (see later in this section).

2. A fully qualified domain name (FQDN) associated (by the Internet DNS service) with an MX record, which MUST be "mailable".

3. An arbitrary name believed to be unique and registered at least with all sites which receive articles directly from the given site.

4. An encoding of an IP address - <IPv4address> or <IPv6address> [RFC 2373] (the requirement to be able to use an <IPv6address> is the reason for including ':' as an allowed character within a path-identity).

The FQDN of an agent is "mailable" if the administrators of that agent can be reached by email using both of the forms "usenet@<FQDN>" and "news@<FQDN>", in conformity with [RFC 2142].

Of the above options, nos. 1 to 3 are much to be preferred, unless there are strong technical reasons dictating otherwise. In

particular, the injecting agent's path-identity MUST, as a special case, be an FQDN as in option 1 or option 2, and MUST be mailable. Additionally, in the case of an injecting agent offering its services to the general public, its administrators MUST also be reachable

using the form "abuse@<FQDN>" UNLESS a more specific complaints address has been specified in a Complaints-To-header (6.20).

[Suggested alternative for 1st two sentences:
For injecting agents, the path-identity MUST be option 1 or 2. For other agents, options 1 through 3 are preferrable.]

The injecting agent's path-identity MUST be followed by the special path-delimiter '%' which serves to separate the pre-injection and post-injection regions of the Path-content (see 5.6.3).

In the case of a relaying or serving agent, the path-delimiter is chosen as follows. When such an agent receives an article, it MUST establish the identity of the source and compare it with the leftmost path-identity of the Path-content. If it matches, a '/' should be used as the path-delimiter when prepending the agent's own path-identity. If it does not match then the agent should prepend two entries to the Path-content; firstly the true established path-identity of the source followed by a '?' path-delimiter, and then, to the left of that, the agent's own path-identity followed by a '/' path-delimiter as usual. This prepending of two entries SHOULD NOT be done if the provided and established identities match.

Any method of establishing the identity of the source may be used with the consideration that, in the event of problems, the agent concerned may be called upon to justify it.

> NOTE: The use of the '%' path-delimiter marks the position of the injecting agent in the chain. In normal circumstances there should therefore be only one '%' path-delimiter present. If more than one '%' is found, then the article has evidently been reinjected (8.2) at some stage, in which case the path-identity in front of the leftmost '%' is to be regarded as the true injecting agent.

5.6.3. The tail-entry

For historical reasons, the tail-entry (i.e. the rightmost entry in the Path-content) is regarded as a "user name", and therefore MUST NOT be interpreted as a site through which the article has already passed. Moreover, the Path-content as a whole is not an email address and MUST NOT be used to contact the poster. Posting and/or injecting agents MAY place any string here. When it is not an actual user name, the string "not-for-mail" is often used.

Often this field will be the only entry in the region (known as the pre-injection region) after the '%', although there may be entries corresponding to machines traversed between the posting agent and the injecting agent proper. In particular, injecting agents that receive articles from many sources MAY include information to establish the

circumstances of the injection such as the identity of the source machine (especially if an Injection-Info-header (6.19) is not being provided). Any such inclusion SHOULD NOT conflict with any genuine site identifier. The '!' path-delimiter may be used freely within the pre-injection region, although '/' and '?' are also appropriate

if used correctly.

5.6.4. Path-Delimiter Summary

A summary of the various path-delimiters. The name immediately to the left of the path-delimiter is always that of the machine which added the path-delimiter.

'/' The name immediately to the right is known to be the identity of the machine from which the article was received (either because the entry was made by that machine and we have verified it, or because we have added it ourselves).

'?' The name immediately to the right is the claimed identity of the machine from which the article was received, but we were unable to verify it (and have prepended our own view of where it came from, and then a '/').

'%' Everything to the right is the pre-injection region followed by the tail-entry. The name on the left is the FQDN of the injecting agent. The presence of two '%'s in a path indicates a reinjection (see 8.2.2).

'!' The name immediately to the right is unverified. The presence of a '!' to the left of the '%' indicates that the identity to the left is that of an old-style system not conformant with this standard.

',' Reserved for future use, treat as '/'.

Other
Old software may possibly use other path-delimiters, which should be treated as '!'. But note in particular that ':', '-' and '_' are components of names, not path-delimiters, and FWS on its own MUST NOT be used as the sole path-delimiter.

NOTE: Old Netnews relaying and injecting agents almost all delimit Path entries with a '!', and these entries are not verified. The presence of '%' indicates that the article was injected by software conforming to this standard, and the presence of '!' to the left of a '%' indicates that the article passed through systems developed prior to this standard. It is anticipated that relaying agents will reject articles in the old style once this new standard has been widely adopted.

5.6.5. Example

```
Path: foo.isp.example/
   foo-server/bar.isp.example?10.123.12.2/old.site.example!
   barbaz/baz.isp.example%dialup123.baz.isp.example!x
```

NOTE: That article was injected into the news stream by
baz.isp.example (complaints may be addressed to
abuse@baz.isp.example). The injector has taken care to record

that it got it from dialup123.baz.isp.example. "x" is a dummy tail-entry, though sometimes a real userid is put there.

The article was relayed, perhaps by UUCP, to the machine known, at least to old.site.example, as "barbaz".

Barbaz relayed it to old.site.example, which does not yet conform to this standard (hence the '!' path-delimiter). So one cannot be sure that it really came from barbaz.

Old.site.example relayed it to a site claiming to have the IP address [10.123.12.2], and claiming (by using the '/' path-delimiter) to have verified that it came from old.site.example.

[10.123.12.2] relayed it to "foo-server" which, not being convinced that it truly came from [10.123.12.2], did a reverse lookup on the actual source and concluded it was known as bar.isp.example (that is not to say that [10.123.12.2] was not a correct IP address for bar.isp.example, but simply that that connection could not be substantiated by foo-server). Observe that foo-server has now added two entries to the Path.

"foo-server" is a locally significant name within the complex site of many machines run by foo.isp.example, so the latter should have no problem recognizing foo-server and using a '/' path-delimiter. Presumably foo.isp.example then delivered the article to its direct clients.

It appears that foo.isp.example and old.site.example decided to fold the line, on the grounds that it seemed to be getting a little too long.

5.7. Injection-Date

The Injection-Date-header contains the date and time that the article was injected into the network. Its purpose is to prevent the reinjection into the news stream of "stale" articles which have already expired by the time they arrive at some relaying or serving agent.

```
header            =/ Injection-Date-header
Injection-Date-header
                  = "Injection-Date" ":" SP Injection-Date-content
                       *( ";" extension-parameter )
Injection-Date-content
                  = date-time
```

See the remarks under the Date-header (5.1) regarding the syntax of a date-time and the requirements and recommendations to which it is subject.

An Injection-Date-header MUST NOT be added to an article except by an injecting agent, hence it will never be present in a proto-article (8.2.1). It MUST be added by each injecting agent but, once added,

it MUST NOT subsequently be changed or removed by any other agent, even during reinjection (8.2.2).

> NOTE: The date-time would normally be expected to be later than the date-time in the Date-header, but differences between the clocks on the various agents and other special circumstances might vitiate that; no provision is made for any such discrepancy to be corrected - better that the injecting agent should just insert the correct time as it sees it.
>
> Since this header is newly introduced in this standard, other agents cannot rely on its always being present; therefore, provision is made (8.3,8.4) for the Date-header to be used when it is absent.

This header is intended to replace the currently-used but nowhere-documented header "NNTP-Posting-Date", whose use is now deprecated.

6. Optional Headers

None of the headers appearing in this section is required to appear in every article but some of them are required in certain types of article, such as followups. Any header defined in this (or any other) standard MUST NOT appear more than once in an article unless specifically stated otherwise. Experimental headers (4.2.5.1) and headers defined by cooperating subnets are exempt from this requirement. See section 8 "Duties of Various Agents" for the full picture.

6.1. Reply-To

The Reply-To-header specifies a reply address(es) to be used for personal replies for the poster(s) of the article when this is different from the poster's address(es) given in the From-header. The content syntax makes use of syntax defined in [RFC 2822].

```
header              =/ Reply-To-header
Reply-To-header     = "Reply-To" ":" SP Reply-To-content
Reply-To-content    = address-list
```

In the absence of Reply-To, the reply address(es) is the address(es) in the From-header.

6.1.1. 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
```

6.2. Sender

The Sender-header specifies the mailbox of the person or entity which caused this article to be posted (and hence injected), if that person or entity is different from that given in the From-header or if more than one mailbox appears in the From-header. This header SHOULD NOT appear in an article unless the sender is different from the poster. This header is appropriate for use by automatic article posters. The content syntax makes use of syntax defined in [RFC 2822].

```
header              =/ Sender-header
Sender-header       = "Sender" ":" SP Sender-content
Sender-content      = mailbox
```

6.3. Organization

The Organization-header is a short phrase identifying the poster's organization.

```
header              =/ Organization-header
Organization-header = "Organization" ":" SP Organization-content
Organization-content= unstructured
```

6.4. 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 content syntax makes use of syntax defined in [RFC 2822].

```
header              =/ Keywords-header
Keywords-header     = "Keywords" ":" SP Keywords-content
Keywords-content    = phrase *( "," phrase )
```

NOTE: The list is comma separated, NOT space separated.

NOTE: Contrary to the usage defined in [RFC 2822], this standard does not permit multiple occurrences of this header.

6.5. Summary

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

```
header              =/ Summary-header
Summary-header      = "Summary" ":" SP Summary-content
Summary-content     = unstructured
```

6.6. Distribution

The Distribution-header is an inheritable header (see 4.2.5.2) which specifies geographical or organizational limits to an article's propagation.

```
header              =/ Distribution-header
Distribution-header = "Distribution" ":" SP Distribution-content
                         *( ";" extension-parameter )
Distribution-content= distribution *( dist-delim distribution )
dist-delim          = ","
distribution        = [FWS] distribution-name [FWS]
distribution-name   = ALPHA 1*distribution-rest
distribution-rest   = ALPHA / "+" / "-" / "_"
```

Articles MUST NOT be passed between relaying agents or to serving agents unless the sending agent has been configured to supply and the receiving agent to receive at least one of the distributions in the Distribution-header. Additionally, reading agents MAY also be configured so that unwanted distributions do not get displayed.

> NOTE: Although it would seem redundant to filter out unwanted distributions at both ends of a relaying link (and it is clearly more efficient to do so at the sending end), many sending sites have been reluctant, historically speaking, to apply such filters (except to ensure that distributions local to their own site or cooperating subnet did not escape); moreover they tended to configure their filters on an "all but those listed" basis, so that new and hitherto unheard of distributions would not be caught. Indeed many "hub" sites actually wanted to receive all possible distributions so that they could feed on to their clients in all possible geographical (or organizational) regions.
>
> Therefore, it is desirable to provide facilities for rejecting unwanted distributions at the receiving end. Indeed, it may be simpler to do so locally than to inform each sending site of what is required, especially in the case of specialized distributions (for example for control messages, such as cancels from certain issuers) which might need to be added at short notice. The possibility for reading agents to filter distributions has been provided for the same reason.

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". However, "world" SHOULD NEVER be mentioned explicitly since it is the default when the Distribution-header is absent entirely. "All" MUST NOT be used as a distribution-name. Distribution-names SHOULD contain at least three characters, except when they are two-letter country names as in [ISO 3166]. Distribution-names are case-insensitive (i.e. "US", "Us" and "us" all specify the same distribution).

Followup agents SHOULD initially supply the same Distribution-header as found in the precursor.

6.7. Followup-To

The Followup-To-header specifies which newsgroup(s) followups should be posted to.

```
header              =/ Followup-To-header
Followup-To-header  = "Followup-To" ":" SP Followup-To-content
                          *( ";" extension-parameter )
Followup-To-content = Newsgroups-content /
                      [FWS] %x70.6F.73.74.65.72 [FWS]
                      ; which is a case-sensitive "poster"
```

The syntax is the same as that of the Newsgroups-content, with the addition that the keyword "poster" is allowed. In the absence of a Followup-To-content, the default newsgroup(s) for a followup are those in the Newsgroups-header.

A Followup-To-header consisting of the keyword "poster" indicates that the poster requests no followups to be sent in response to this article, only personal replies to the article's reply address. Although the keyword "poster" is case-sensitive, followup agents MAY choose to regognize case insensitive forms such as "Poster".

> NOTE: A poster who wishes both a personal reply and a followup post should include an appropriate Mail-Copies-To-header (6.8).

6.8. Mail-Copies-To

The Mail-Copies-To-header indicates whether or not the poster wishes to have followups to an article emailed in addition to being posted to Netnews and, if so, establishes the address to which they should be sent.

The content syntax makes use of syntax defined in [RFC 2822].

```
header      =/ Mail-Copies-To-header
Mail-Copies-To-header
            = "Mail-Copies-To" ":" SP Mail-Copies-To-content
Mail-Copies-To-content
            = copy-addr / [CFWS] ( "nobody" / "poster" ) [CFWS]
copy-addr   = address-list
```

The keyword "nobody" indicates that the poster does not wish copies of any followup postings to be emailed. This indication is widely seen as a very strong wish, and is to be taken as the default when this header is absent.

The keyword "poster" indicates that the poster wishes a copy of any followup postings to be emailed to him.

Otherwise, this header contains a copy-addr to which the poster wishes a copy of any followup postings to be sent.

NOTE: Some existing practice uses the keyword "never" in place of "nobody" and "always" in place of "poster". These usages are deprecated, but followup agents MAY observe them. The actions to be taken by by followup agent when following up to an article containing a Mail-Copies-To header are set out in section 8.6.

Whether or not this header will also find similar usage for replies to messages sent to mailing lists falls outside the scope of this standard.

6.9. Posted-And-Mailed

```
header      =/ Posted-And-Mailed-header
Posted-And-Mailed-header
            = "Posted-And-Mailed" ":" SP Posted-And-Mailed-content
                 *( ";" extension-parameter )
Posted-And-Mailed-content
            = [CFWS] ( "yes" / "no" ) [CFWS]
```

This header, when used with the "yes" keyword, indicates that the article has been both posted to the specified newsgroups and emailed. It SHOULD be used when replying to the poster of an article to which this one is a followup (see the Mail-Copies-To-header in section 6.8) and it MAY be used when any article is also mailed to a recipient(s) identified in a To- and/or Cc-header that is also present. The "no" keyword is included for the sake of completeness; it MAY be used to indicate the opposite state, but is redundant insofar as it only describes the default state when this header is absent.

This header, if present, MUST be included in both the posted and emailed versions of the article. The Newsgroups-header of the posted article SHOULD be included in the email version as recommended in section 5.5. All other headers defined in this standard (excluding variant headers) MUST be identical in both the posted and emailed versions of the article. The bodies MUST be identical in both, apart from a possible change of Content-Transfer-Encoding.

NOTE: This leaves open the question of whether a To- or a Cc-header should appear in the posted version. Naturally, a Bcc-header should not appear, except in a form which indicates that there are additional unspecified recipients.

6.10. References

The References-header is an inheritable header (see 4.2.5.2) which lists CFWS-separated message identifiers of the article's precursors, as described in 8.6. The content syntax makes use of syntax defined in [RFC 2822] (but see the revised definition of msg-id in section 2.4.3).

```
header              =/ References-header
References-header   = "References" ":" SP References-content
References-content  = [CFWS] msg-id *( CFWS msg-id ) [CFWS]
```

NOTE: This differs from the syntax of [RFC 2822] by requiring at least one CFWS between the msg-ids (a SP at this point was an [RFC 1036] requirement).

A followup MUST have a References-header, and an article that is not a followup MUST NOT have a References-header.

6.10.1. 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>
```

6.11. Expires

The Expires-header specifies a date and time when the article is deemed to be no longer relevant and could usefully be removed ("expired"). The content syntax makes use of syntax defined in [RFC 2822].

```
header              =/ Expires-header
Expires-header      = "Expires" ":" SP Expires-content
                        *( ";" extension-parameter )
Expires-content     = date-time
```

NOTE: This header is suitable for specifying both unusually short and unusually long expiry times; there is little point in using it in other circumstances.

6.12. Archive

This optional header provides an indication of the poster's intent regarding preservation of the article in publicly accessible long-term or permanent storage.

```
header              =/ Archive-header
Archive-header      = "Archive" ":" SP Archive-content
                        *( ";" ( Archive-parameter /
                                 extension-parameter ) )
Archive-content     = [CFWS] ("no" / "yes" ) [CFWS]
Archive-parameter   = <a parameter with attribute "filename"
                       and any value>
```

The presence of an "Archive: no" header in an article indicates that the poster does not permit redistribution from publicly accessible long-term or permanent archives. The absence of this header, or an

explicit "Archive: yes", indicates that the poster is willing for such redistribution to take place. The optional "filename" parameter can then be used to suggest a filename under which the article should be stored. Further extensions to this standard may provide additional

parameters for administration of the archiving process.

NOTE: This standard does not attempt to define the length of "long-term", since it is dependent on many factors, including the retention policies of individual sites, and the customs or policies established for particular newsgroups or hierarchies.

NOTE: Posters are cautioned that some sites may not implement the "no" option of the Archive-header correctly. In some jurisdictions non-compliance with this header may constitute a breach of copyright or of other legal provisions. Moreover, even if this header prevents the poster's words from being archived publicly, it does nothing to prevent the archiving of a followup in which those words are quoted.

6.13. Control

The Control-header marks the article as a control message, and specifies the desired actions (additional to the usual ones of storing and/or relaying the article).

```
header              =/ Control-header
Control-header      = "Control" ":" SP Control-content
                         *( ";" extension-parameter )
Control-content     = [CFWS] control-message [CFWS]
control-message     = <empty>
```

However, the rule given above for control-message is incomplete. Further alternatives will be added incrementally as the various control-messages are introduced in section 7, or in extensions to this standard, using the "=/" notation defined in [RFC 2234]. For example, a putative "Control-message" would be defined as follows:

```
control-message     =/ Control-message
Control-message     = "Control" Control-arguments
Control-arguments   = <the argument(s) specific to that
                       Control-message>
```

where "Control" is a "verb" which is (and MUST be) of the syntactic form of a token and Control-arguments MUST be of the syntactic form of a CFWS-separated list of values (which may require the use of quoted-strings if any tspecials or non-ASCII characters are involved).

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.

An article with a Control-header MUST NOT also have a Supersedes-header.

NOTE: The presence of a Subject-header starting with the string "cmsg " and followed by a Control-message MUST NOT be construed, in the absence of a proper Control-header, as a request to

perform that control action (as may have occurred in some legacy software).

6.14. Approved

The Approved-header indicates the mailing addresses (possibly including the full names) of the persons or entities approving the article for posting.

```
header              =/ Approved-header
Approved-header     = "Approved" ":" SP Approved-content
                         *( ";" extension-parameter )
Approved-content    = From-content  ; see 5.2
```

Each mailbox contained in the Approved-content MUST be that of one of the person(s) or entity(ies) in question, and one of those mailboxes MUST be that of the actual injector of the article.

An Approved-header is required in all postings to moderated newsgroups. If this header is not present in such postings, then serving agents MUST (and relaying agents MAY) reject the article. Please see section 8.2.2 for how injecting agents should treat postings to moderated groups that do not contain this header.

An Approved-header is also required in certain control messages, to reduce the risks of accidental or unauthorized posting of same.

> NOTE: The presence of an Approved-header indicates that the person or entity identified claims to have the necessary authority to post the article in question, thus enabling sites that dispute that authority to refuse to accept or to act upon it. However, the mere presence of the header is insufficient to provide assurance that it indeed originated from that person or entity, and it is therefore desirable that it be included within some digital signature scheme (see 7.1), especially in the case of control messages (section 7).

6.15. Supersedes

The Supersedes-header contains a message identifier specifying an article to be superseded upon the arrival of this one. The specified article MUST be treated as though a "cancel" control message had arrived for the article (but observe that a site MAY choose not to honour a "cancel" message, especially if its authenticity is in doubt). The content syntax makes use of syntax defined in [RFC 2822] (but see the revised definition of msg-id in section 2.4.3).

```
header              =/ Supersedes-header
Supersedes-header   = "Supersedes" ":" SP Supersedes-content
Supersedes-content  = [CFWS] msg-id [CFWS]
```

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

NOTE: The Supersedes-header defined here has no connection with the Supersedes-header that sometimes appears in Email messages converted from X.400 according to [RFC 2156]; in particular, the syntax here permits only one msg-id in contrast to the multiple msg-ids in that Email version.

Thus when an article contains a Supersedes-header, the old article mentioned SHOULD be withdrawn from circulation or access, as in a cancel message (7.3), and the new article inserted into the system as any other new article would have been.

Whatever security or authentication checks are normally applied to a Control cancel message (or may be prescribed for such messages by some extension to this standard - see the remarks in 7.1 and 7.3) MUST also be applied to an article with a Supersedes-header. In the event of the failure of such checks, the article SHOULD be discarded, or at most stored as an ordinary article.

6.16. Xref

The Xref-header is a variant header (4.2.5.3) which indicates where an article was filed by the last serving agent to process it.

```
header            =/ Xref-header
Xref-header       = "Xref" ":" SP Xref-content
                       *( ";" extension-parameter )
Xref-content      = [CFWS] server-name 1*( CFWS location ) [CFWS]
server-name       = path-identity  ; see 5.6.1
location          = newsgroup-name ":" article-locator
article-locator   = 1*( %x21-27 / %x29-3A / %x3C-7E )
                       ; US-ASCII printable characters
                       ; except '(' and ';'
```

The server-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 much other software 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 MAY delete it before relaying, but otherwise it SHOULD be ignored by any relaying or serving agent receiving it.

It is convenient, though not required, for a serving agent to use the same server-name in Xref-headers as the path-identity it uses in Path-headers (just so long as reading agents can distinguish it from other serving agents known to them).

6.17. Lines

The Lines-header indicates the number of lines in the body of the article.

```
header              =/ Lines-header
Lines-header        = "Lines" ":" SP Lines-content
                         *( ";" extension-parameter )
Lines-content       = [CFWS] 1*DIGIT [CFWS]
```

The line count includes all body lines, including the signature if any, including empty lines (if any) at the beginning or end of the body, and including the whole of all MIME message and multipart parts contained in 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.

This header is to be regarded as obsolete, and it will likely be removed entirely in a future version of this standard. In the meantime, its use is deprecated.

6.18. User-Agent

The User-Agent-header contains information about the user agent (typically a newsreader) generating the article, for statistical purposes and tracing of standards violations to specific software needing correction. Although not one of the mandatory headers, posting agents SHOULD normally include it. It is also intended that this header be suitable for use in Email.

```
header              =/ User-Agent-header
User-Agent-header   = "User-Agent" ":" SP User-Agent-content
                         *( ";" extension-parameter )
User-Agent-content  = product *( CFWS product )
product             = [CFWS] token [CFWS] [ "/" product-version ]
product-version     = [CFWS] token [CFWS]
```

This header MAY contain multiple product-tokens identifying the agent and any subproducts which form a significant part of the posting agent, listed in order of their significance for identifying the application. Product-tokens should be short and to the point - they MUST NOT be used for information beyond the canonical name of the product and its version. Injecting agents MAY include product information for themselves (such as "INN/1.7.2"), but relaying and serving agents MUST NOT generate or modify this header to list themselves.

> NOTE: Minor variations from [RFC 2616] which describes a similar facility for the HTTP protocol:

1. "{" and "}" are allowed in a token (product and product-version) in Netnews,

2. Comments are permitted wherever whitespace is allowed.

NOTE: This header supersedes the role performed redundantly by experimental headers such as X-Newsreader, X-Mailer, X-Posting-Agent, X-Http-User-Agent, and other headers previously used on Usenet and in Email for this purpose. Use of these experimental headers SHOULD be discontinued in favor of the single, standard User-Agent-header.

6.18.1. Examples

```
User-Agent: tin/1.3-950621beta-PL0 (Unix)
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: inn/1.7.2
User-Agent: telnet
```

6.19. Injection-Info

The Injection-Info-header SHOULD be added to each article by the injecting agent in order to provide information as to how that article entered the Netnews system and to assist in tracing its true origin.

```
header          =/ Injection-Info-header
Injection-Info-header
                = "Injection-Info" ":" SP Injection-Info-content
                     *( ";" ( Injection-Info-parameter /
                              extension-parameter ) )
Injection-Info-content
                = [CFWS] path-identity [CFWS]
Injection-Info-parameter
                = posting-host-parameter /
                  posting-account-parameter /
                  posting-sender-parameter /
                  posting-logging-parameter
                  ; for {USENET}-parameters see 4.1
posting-host-parameter
                = <a parameter with attribute "posting-host"
                   and value some host-value>
host-value      = dot-atom /
                  [ dot-atom ":" ]
                    ( IPv4address / IPv6address ); see [RFC 2373]
posting-account-parameter
                = <a parameter with attribute "posting-account"
                   and any value>
posting-sender-parameter
```

```
                    = <a parameter with attribute "sender"
                       and value some sender-value>
      sender-value    = mailbox / "verified"
```

```
   posting-logging-parameter
                  = <a parameter with attribute "logging-data"
                     and any value>
```

An Injection-Info-header MUST NOT be added to an article except by an injecting agent. Any Injection-Info-header present when an article arrives at an injecting agent MUST be removed. In particular if, for some exceptional reason, an article is being reinjected (8.2.2), the Injection-Info-header will always relate to the latest injection (to the extent that this happens, the Injection-Info-header can be regarded as a variant header).

The path-identity MUST be the same as the path-identity prepended to the Path-header by that same injecting agent which, following section 5.6.2, MUST therefore be a fully qualified domain name (FQDN) mailable address.

Although comments and folding of white space are permitted throughout the Injection-Info-content specification, it is RECOMMENDED that folding is not used within any parameter (but only before or after the ";" separating those parameters), and that comments are only used following the last parameter. It is also RECOMMENDED that such parameters as are present are included in the order in which they have been defined in the syntax above. An injecting agent SHOULD use a consistent form of this header for all articles emanating from the same or similar origins.

> NOTE: The effect of those recommendations is to facilitate the recognition of articles arising from certain designated origins (as in the so-called "killfiles" which are available in some reading agents). Observe that the order within the syntax has been chosen to place last those parameters which are most likely to change between successive articles posted from the same origin.
>
> NOTE: To comply with the overall "attribute = value" syntax of parameters, any value containing an IPv6address, a date-time, a mailbox, or any CFWS MUST be quoted using <DQUOTE>s (the quoting is optional in other cases).
>
> NOTE: This header is intended to replace various currently-used but nowhere-documented headers such as "NNTP-Posting-Host" and "X-Trace". These headers are now deprecated, and any of them present when an article arrives at an injecting agent SHOULD also be removed as above.

6.19.1. Usage of Injection-Info-parameters

The purpose of these parameters is to enable the injecting agent to

make assertions about the origin of the article, in fulfilment of its responsibilities towards the rest of the network as set out in section 8.2.

An injecting agent MUST NOT include any Injection-Info-parameter unless it has positive evidence of its correctness. An injecting agent MAY also include further extension-parameters with x-attributes which will assist in identifying the origin of the article.

6.19.1.1. The posting-host-parameter

If a dot-atom is present, it MUST be a FQDN identifying the specific host from which the injecting agent received the article. Alternatively, an IP address (IPv4address or IPv6address) identifies that host. If both forms are present, then they MUST identify the same host, or at least have done so at the time the article was injected.

> NOTE: It is commonly the case that this parameter identifies a dial-up point-of-presence, in which case a posting-account or logging-data may need to be consulted to find the true origin of the article.

6.19.1.2. The posting-account-parameter

This parameter identifies the source from which the injecting agent received the article. It SHOULD be in a cryptic notation understandable only by the administrator of the injecting agent, but it MUST be such that a given source gives rise to the same posting-account, at least in the short term. If the injecting agent is unable to meet that obligation, then it should use a posting-logging-parameter instead.

6.19.1.3. The posting-sender-parameter

This parameter identifies the mailbox of the verified sender of the article (alternatively, it uses the token "verified" to indicate that at least any addr-spec in the Sender-header of the article, or in the From-header if the Sender-header is absent, is correct).

> NOTE: An injecting agent is unlikely to be able to make use of this parameter except in cases where it is running on a machine which is aware of the user-space in which the posting agent is operating. This parameter should be used in preference to a posting-account-parameter in such situations.

6.19.1.4. The posting-logging-parameter

This parameter contains information (typically a session number or other non-persistent means of identifying a posting account) which will enable the true origin of the article to be determined by reference to logging information kept by the injecting agent.

6.19.2. Example

```
      Injection-Info: news2.isp.net; posting-host=modem-15.pop.isp.net;
         posting-account=client0002623; logging-data=2427"
```

6.20. Complaints-To

The Complaints-To-header is added to an article by an injecting agent in order to indicate the mailbox to which complaints concerning the poster of the article may be sent. The content syntax makes use of syntax defined in [RFC 2822].

```
header            =/ Complaints-To-header
Complaints-To-header
                  = "Complaints-To" ":" SP Complaints-To-content
Complaints-To-content
                  = address-list
```

A Complaints-To-header MUST NOT be added to an article except by an injecting agent. Any Complaints-To-header present when an article arrives at an injecting agent MUST be removed. In particular if, for some exceptional reason an article is being reinjected (8.2.2), the Complaints-To-header will always relate to the latest injection (to the extent that this happens, the Complaints-To-header can be regarded as a variant header).

The specified mailbox is for sending complaints concerning the behaviour of the poster of the article; it SHOULD NOT be used for matters concerning propagation, protocol problems, etc. which should be addressed to "usenet@" or "news@" the path-identity which was prepended to the Path-header by the injecting agent, in accordance with section 5.6.2. In the absence of this header, complaints concerning a poster's behaviour MAY be addressed to "abuse@" that path-identity (although section 5.6.2 provides no obligation for that address to be mailable at an injecting agent that is not provided for the use of the general public).

6.21. MIME headers

6.21.1. Syntax

The following headers may be used within articles conforming to this standard.

MIME-Version:	[RFC 2045]
Content-Type:	[RFC 2045],[RFC 2046]
Content-Transfer-Encoding:	[RFC 2045]
Content-ID:	[RFC 2045]
Content-Description:	[RFC 2045]
Content-Disposition:	[RFC 2183]
Content-Location:	[RFC 2557]
Content-Language:	[RFC 3282]
Content-MD5:	[RFC 1864]

The RFCs listed are deemed to be incorporated into this standard to

the extent necessary to facilitate their usage within Netnews, subject to curtailment of that usage as described in the following sections. Moreover, extensions to those standards registered in accordance with [RFC 2048] are also available for use within Netnews,

as indeed is any other header in the Content-* series which has a sensible interpretation within Netnews.

Insofar as the syntax for these headers, as given in those RFCs does not specify precisely where whitespace and comments may occur (whether in the form of WSP, FWS or CFWS), the usage defined in this standard, and failing that in [RFC 2822], and failing that in [RFC 822] MUST be followed. In particular, there MUST NOT be any WSP between a header-name and the following colon and there MUST be a SP following that colon.

6.21.2. Content-Type

If the contents of an article is something other than plain text in the US-ASCII charset (these being the default assumptions), an appropriate Content-Type-header MUST be included.

Reading agents SHOULD NOT, unless explicitly configured otherwise, act automatically on Application types which could change the state of that agent (e.g. by writing or modifying files), except in the case of those prescribed for use in control messages (7.2.1.2 and 7.2.4.1).

The Content-Type "message/partial" MAY be used to split a long news article into several smaller ones. However, breaking long texts into several parts is usually unnecessary, since modern transport agents should have no difficulty in handling articles of arbitrary length, although it may be useful to break long binaries.

IF the Content-Type "message/partial" is used, then the "id" parameter SHOULD be in the form of a unique message identifier (but different from that in the Message-ID-header of any of the parts). The second and subsequent parts SHOULD contain References-headers referring to all the previous parts, thus enabling reading agents with threading capabilities to present them in the correct order. Reading agents MAY then provide a facility to recombine the parts into a single article (but this standard does not require them to do so).

The Content-Type "message/rfc822" should be used for the encapsulation (whether as part of another news article or, more usually, as part of an email message) of complete news articles which have already been posted to Netnews and which are for the information of the recipient, and do not constitute a request to repost them (refer to 6.21.4.2 for the now obsolete "message/news" formerly intended for this purpose).

6.21.3. Content-Transfer-Encoding

"Content-Transfer-Encoding: 7bit" is sufficient for article bodies

(or parts of multiparts) written in pure US-ASCII (or most other material representable in 7 bits). Posting agents SHOULD specify "Content-Transfer-Encoding: 8bit" for all other cases except where the content is (or might be) "8bit-unsafe", or where some protocol

explicitly disallows it. They MAY use "8bit" encoding even when "7bit" encoding would have sufficed.

Content is "8bit-unsafe" if it contains octets equivalent to the US-ASCII characters CR or LF (other than in the combination CRLF) or NUL. This is often the case with application types (though in many cases application types are intended to be human readable, in which case they will usually be 8bit-safe). It also arises with certain charsets (as indicated in the Content-Type-header), particularly in the case of 16-bit charsets such as UTF-16 ([UNICODE 3.2] or [ISO/IEC 10646]).

Examples of protocols REQUIRING particular Content-Transfer-Encodings include the Content-Type "application/pgp-signature" [RFC 3156], and the Content-Type "message/partial" which itself MUST use "Content-Transfer-Encoding: 7bit" (though the encapsulated complete message may itself use encoding "quoted-printable" or "base64", but that information is only conveyed along with the first of the partial parts).

Encoding "binary" MUST NOT be used (except in cooperating subnets with alternative transport arrangements) because this standard does not mandate a transport mechanism that could support it.

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

6.21.4. Definition of some new Content-Types

This standard defines (or redefines) several new Content-Types, which require to be registered with IANA as provided for in [RFC 2048]. For "application/news-groupinfo" see 7.2.1.2, for "application/news-checkgroups" see 7.2.4.1, and for "application/news-transmission" see the following section.

6.21.4.1. Application/news-transmission

The Content-Type "application/news-transmission" is intended for the encapsulation of complete news articles where the intention is that the recipient should then inject them into Netnews. This Application type provides one of the methods for mailing articles to moderators (see 8.2.2) and it is also the preferred method when sending to an email-to-news gateway (see 8.8.2).

> NOTE: The benefit of such encapsulation is that it removes possible conflict between news and email headers and it provides a convenient way of "tunnelling" a news article through a transport medium that does not support 8bit characters.

The MIME content type definition of "application/news-transmission" is:

MIME type name: application

MIME subtype name:	news-transmission
Required parameters:	none
Optional parameters:	usage=moderate usage=inject usage=relay
Encoding considerations:	A transfer-encoding (such as Quoted-Printable or Base64) different from that of the article transmitted MAY be supplied (perhaps en route) to ensure correct transmission over some 7bit transport medium.
Security considerations:	A news article may be a "control message", which could have effects on the recipient host's system beyond just storage of the article. However, such control messages also occur in normal news flow, so most hosts will already be suitably defended against undesired effects.
Published specification:	[USEFOR]
Body part:	A complete article or proto-article, ready for injection into Netnews, or a batch of such articles.

> NOTE: It is likely that the recipient of an "application/news-transmission" will be a specialized gateway (e.g. a moderator's submission address) able to accept articles with only one of the three usage parameters "moderate", "inject" and "relay", hence the reason why they are optional, being redundant in most situations. Nevertheless, they MAY be used to signify the originator's intention with regard to the transmission, so removing any possible doubt.

When the parameter "relay" is used, or implied, the body part MAY be a batch of articles to be transmitted together, in which case the following syntax MUST be used.

```
batch             = 1*( batch-header article )
batch-header      = "#!" SP rnews SP article-size CRLF
rnews             = %x72.6E.65.77.73 ; case sensitive "rnews"
article-size      = 1*DIGIT
```

Thus a batch is a sequence of articles, each prefixed by a header line that includes its size. The article-size is a decimal count of the octets in the article, counting each CRLF as one octet regardless of how it is actually represented.

> NOTE: Despite the similarity of this format to an executable UNIX script, it is EXTREMELY unwise to feed such a batch into a command interpreter in anticipation of it running a command

named "rnews"; the security implications of so doing would be disastrous.

6.21.4.2. Message/news obsoleted

The Content-Type "message/news", as previously registered with IANA, is hereby declared obsolete. It was never widely implemented, and its default treatment as "application/octet-stream" by agents that did not recognize it was counter productive. The Content-Type "message/rfc822" SHOULD be used in its place, as already described above.

6.22. Obsolete Headers

Persons writing new agents SHOULD ignore any former meanings of the following headers:

Also-Control
See-Also
Article-Names
Article-Updates

7. Control Messages

The following sections document the control messages. "Message" is used herein as a synonym for "article" unless context indicates otherwise.

The Newsgroups-header of each control message SHOULD include the newsgroup-name(s) for the group(s) affected (i.e. groups to be created, modified or removed, or containing articles to be canceled). This is to ensure that the message propagates to all sites which receive (or would receive) that group(s). It MAY include other newsgroup-names so as to improve propagation (but this practice may cause the control message to propagate also to places where it is unwanted, or even cause it not to propagate where it should, so it should not be used without good reason).

The descriptions below set out REQUIREMENTS to be followed by sites that receive control messages and choose to honour them. However, nothing in these descriptions should be taken as overriding the right of any such site, in accordance with its local policy, to deny any particular control message, or to refer it to an administrator for approval (either as a class or on a case-by-case basis).

Relaying Agents MUST propagate all control messages regardless of whether or not they are recognized or processed locally.

In the following sections, each type of control message is defined syntactically by defining its verb, its arguments, and possibly its body.

7.1. Digital Signature of Headers

It is most desirable that group control messages (7.2) in particular be authenticated by incorporating them within some digital signature scheme that encompasses other headers closely associated with them

(including at least the Approved-, Message-ID- and Date-headers). At the time of writing, this is usually done by means of a protocol known as "PGPverify" ([PGPVERIFY]), and continued usage of this is encouraged at least as an interim measure.

However, PGPverify is not considered suitable for standardization in its present form, for various technical reasons. It is therefore expected that an early extension to this standard will provide a robust and general purpose digital authentication mechanism with applicability to all situations requiring protection against malicious use of, or interference with, headers. That extension would also address other Netnews security issues.

7.2. Group Control Messages

"Group control messages" are the sub-class of control messages that request some update to the configuration of the groups known to a serving agent, namely "newgroup". "rmgroup", "mvgroup" and "checkgroups", plus any others created by extensions to this standard.

All of the group control messages MUST have an Approved-header (6.14) which, in those hierarchies where appropriate administrative agencies exist (see 1.1), identifies the appropriate person or entity as authorized by those agencies. The authorized person or entity SHOULD adhere to the conventions and restrictions on the format of newsgroup-names established for those hierarchies (see 5.5).

7.2.1. The 'newgroup' Control Message

```
control-message     =/ Newgroup-message
Newgroup-message    = "newgroup" Newgroup-arguments
Newgroup-arguments  = CFWS newsgroup-name [ CFWS newgroup-flag ]
newgroup-flag       = "moderated"
```

The "newgroup" control message requests that the specified group be created or changed. If the request is honoured, or if the group already exists on the serving agent, and if the newgroup-flag "moderated" is present, then the group MUST be marked as moderated, and vice versa. "Moderated" is the only such flag defined by this standard; other flags MAY be defined for use in cooperating subnets, but newgroup messages containing them MUST NOT be acted on outside of those subnets.

> NOTE: Specifically, some alternative flags such as "y" and "m", which are sent and recognized by some current software, are NOT part of this standard. Moreover, some existing implementations treat any flag other than "moderated" as indicating an unmoderated newsgroup. Both of these usages are contrary to this

standard and control messages with such non-standard flags should be ignored.

The message body comprises or includes an "application/news-groupinfo" (7.2.1.2) part containing machine- and human-readable information about the group.

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

7.2.1.1. The Body of the 'newgroup' Control Message

The body of the newgroup message contains the following subparts, preferably in the order shown:

1. An "application/news-groupinfo" part (7.2.1.2) containing the name and newsgroups-line of the group(s). This part MUST be present.

2. Other parts containing useful information about the background of the newgroup message (typically of type "text/plain").

3. Parts containing initial articles for the newsgroup. See section 7.2.1.3 for details.

In the event that there is only the single (i.e. application/news-groupinfo) subpart present, it will suffice to include a "Content-Type: application/news-groupinfo" amongst the headers of the control message. Otherwise, a "Content-Type: multipart/mixed" header will be needed, and each separate part will then need its own Content-Type-header.

7.2.1.2. Application/news-groupinfo

The "application/news-groupinfo" body part contains brief information about a newsgroup, i.e. the group's name, it's newsgroup-description and the moderation-flag.

> NOTE: The presence of the newsgroups-tag "For your newsgroups file:" is intended to make the whole newgroup message compatible with current practice as described in [Son-of-1036].

The MIME content type definition of "application/news-groupinfo" is:

MIME type name:	application
MIME subtype name:	news-groupinfo
Required parameters:	none
Disposition:	by default, inline
Encoding considerations:	"7bit" or "8bit" is sufficient and MUST be used to maintain compatibility.
Security considerations:	this type MUST NOT be used except as part of a control message for the creation or modification of a Netnews newsgroup
Published specification:	[USEFOR]

The content of the "application/news-groupinfo" body part is defined as:

```
groupinfo-body      = [ newsgroups-tag CRLF ]
                          newsgroups-line CRLF
newsgroups-tag      = %x46.6F.72 SP %x79.6F.75.72 SP
                          %x6E.65.77.73.67.72.6F.75.70.73 SP
                          %x66.69.6C.65.3A
                          ; case sensitive
                          ; "For your newsgroups file:"
newsgroups-line     = newsgroup-name
                          [ 1*HTAB newsgroup-description ]
                          [ 1*WSP moderation-flag ]
newsgroup-description
                    = utext *( *WSP utext )
moderation-flag     = %x28.4D.6F.64.65.72.61.74.65.64.29
                          ; case sensitive "(Moderated)"
```

The newsgroup-description MUST NOT contain any occurrence of the string "(Moderated)" within it.

The "application/news-groupinfo" is used in conjunction with the "newgroup" (7.2.1) and "mvgroup" (7.2.3) control messages. The newsgroup-name in the newsgroups-line MUST agree with the newsgroup-name in the "newgroup" or "mvgroup" control message. The Content-Type "application/news-groupinfo" MUST NOT be used except as a part of such control messages. Although optional, the newsgroups-tag SHOULD be included until such time as this standard has been widely adopted, to ensure compatibility with present practice.

Moderated newsgroups MUST be marked by appending the case sensitive text " (Moderated)" at the end. It is NOT recommended that the moderator's email address be included in the newsgroup-description as has sometimes been done.

> NOTE: There is no provision for the use of charsets other than US-ASCII within a newsgroup-description. Such a facility may be provided in a future extension to this standard.

[That may seem harsh, but if we make any such provision now, it will make life more complicated and restrict our freedom when it comes to the proposed I18N extension. Therefore I resisted the temptation to include any charset parameter with this Content-Type. Note that this also applies to the checkgroups message further on.]

7.2.1.3. Initial Articles

Some subparts of a "newgroup" or "mvgroup" control message MAY contain an initial set of articles to be posted to the affected newsgroup(s) as soon as it has been created or modified. These parts are identified by having the Content-Type "application/news-transmission", possibly with the parameter "usage=inject". The body of each such part should be a complete proto-article, ready for

posting. This feature is intended for the posting of charters, initial FAQs and the like to the newly formed group(s).

The Newsgroups-header of the proto-article MUST include the newsgroup-name of the newly created or modified group. It MAY include other newsgroup-names. If the proto-article includes a Message-ID-header, the message identifier in it MUST be different from that of any existing article and from that of the control message as a whole. Alternatively such a message identifier MAY be derived by the injecting agent when the proto-article is posted. The proto-article SHOULD include the header "Distribution: local".

The proto-article SHOULD be injected at the serving agent that processes the control message AFTER the newsgroup in question has been created or modified. It MUST NOT be injected if the newsgroup is not, in fact, created (for whatever reason). It MUST NOT be submitted to any relaying agent for transmission beyond the serving agent(s) upon which the newsgroup creation has just been effected (in other words, it is to be treated as having a "Distribution: local" header, whether such a header is actually present or not).

> NOTE: It is not precluded that the proto-article is itself a control message or other type of special article, to be activated only upon creation of the new newsgroup. However, except as might arise from that possibility, any "application/news-transmission" within some nested "multipart/*" structure within the proto-article is not to be activated.

7.2.1.4. Example

A "newgroup" with its charter:

```
From: "example.all Administrator" <admin@noc.example>
Newsgroups: example.admin.info,example.admin.announce
Date: 27 Feb 2002 12:50:22 +0200
Subject: cmsg newgroup example.admin.info moderated
Approved: admin@noc.example
Control: newgroup example.admin.info moderated
Message-ID: <ng-example.admin.info-20020227@noc.example>
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="nxtprt"
Content-Transfer-Encoding: 8bit

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

For your newsgroups file:
example.admin.info      About the example.* groups (Moderated)

--nxtprt
Content-Type: application/news-transmission
```

```
Newsgroups: example.admin.info
From: "example.all Administrator" <admin@noc.example>
Subject: Charter for example.admin.info
Message-ID: <charter-example.admin.info-20020227@noc.example>
```

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

The group example.admin.info contains regularly posted
information on the example.* hierarchy.

--nxtprt--
```

7.2.2. The 'rmgroup' Control Message

```
control-message     =/ Rmgroup-message
Rmgroup-message     = "rmgroup" Rmgroup-arguments
Rmgroup-arguments   = CFWS newsgroup-name
```

The "rmgroup" control message requests that 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 explanatory text.

> NOTE: It is entirely proper for a serving agent to retain the group until all the articles in it have expired, provided that it ceases to accept new articles.

7.2.2.1. Example

```
From: "example.all Administrator" <admin@noc.example>
Newsgroups: example.admin.obsolete, example.admin.announce
Date: 4 Apr 2002 22:04 -0900 (PST)
Subject: cmsg rmgroup example.admin.obsolete
Message-ID: <rm-example.admin.obsolete-20020404@noc.example>
Approved: admin@noc.example
Control: rmgroup example.admin.obsolete

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

7.2.3. The 'mvgroup' Control Message

```
control-message   =/ Mvgroup-message
Mvgroup-message   = "mvgroup" Mvgroup-arguments
Mvgroup-arguments = CFWS newsgroup-name CFWS newsgroup-name
                        [ CFWS newgroup-flag ]
```

The "mvgroup" control message requests that the group specified by the first (old-)newsgroup-name be moved to that specified by the second (new-)newsgroup-name. Thus it is broadly equivalent to a "newgroup" control message for the second group followed by a "rmgroup" control message for the first group.

The message body contains an "application/news-groupinfo" part

(7.2.1.2) containing machine- and human-readable information about the new group, and possibly other subparts as for a "newgroup" control message. The information conveyed in the "application/news-groupinfo" body part, notably its newsgroups-line (7.2.1.2), is

applied to the new group.

When this message is received, the new group is created (if it does not exist already) as for a "newgroup" control message, and MUST in any case be made moderated if a newgroup-flag "moderated" is present, and vice versa. At the same time, arrangements SHOULD be made to remove the old group (as with a "rmgroup" control message), but only after a suitable overlap period to allow the network to adjust to the new arrangement.

At the same time as a serving agent acts upon this message, all injecting agents associated with that serving agent SHOULD inhibit the posting of new articles to the old group (preferably with some indication to the poster that the new group should have been used). Relaying agents, however, MUST continue to propagate such articles during the overlap period.

> NOTE: It is to be expected that different serving agents will act on this message at different points of time, users of the old group will have to become accustomed to the new arrangement, and followups to already established threads will likely continue under the old group. Therefore, there needs to be an overlap period during which articles may continue to be accepted by relaying and serving agents in either group. This standard does not specify any standard period of overlap (though it would be expected to be expressed in days rather than in months). The inhibition of injection of new articles to the old group may seem draconian, but it is the surest way to prevent the changeover from dragging on indefinitely.

Since the "mvgroup" control message is newly introduced in this standard and may not be widely implemented initially, it SHOULD be followed shortly afterwards by a corresponding "newgroup" control message; and again, after a reasonable overlap period, it MUST be followed by a "rmgroup" control message for the old group.

In order to facilitate a smooth changeover, serving agents MAY arrange to service requests for access to the old group by providing access to the new group, which would then contain, or appear to contain, all articles posted to either group (including, ideally, the pre-changeover articles from the old one). Nevertheless, if this feature is implemented, the articles themselves, as supplied to reading agents, MUST NOT be altered in any way (and, in particular, their Newsgroups-headers MUST contain exactly those newsgroups present when they were injected). On the other hand, the Xref-header MAY contain entries for either group (or even both).

> NOTE: Some serving agents that use an "active" file permit an entry of the form "oldgroup xxx yyy =newgroup", which enables

any articles arriving for oldgroup to be diverted to newgroup, thus providing a simple implementation of this feature. However, it is known that not all current serving agents will find implementation so easy (especially in the short term) which is why it is not mandated by this standard. Nevertheless, its

eventual implementation in all serving agents is to be considered highly desirable.

On the other hand, it is recognized that this feature would likely not be implementable if the new group was already in existence with existing articles in it. This situation should not normally arise except when there is already some confusion as to which groups are, or are not, supposed to exist in that hierarchy. Note that the "mvgroup" control message is not really intended to be used for merging two existing groups.

7.2.3.1. Example

```
From: "example.all Administrator" <admin@noc.example>
Newsgroups: example.oldgroup,example.newgroup,example.admin.announce
Date: 30 Apr 2002 22:04 -0500 (EST)
Subject: cmsg mvgroup example.oldgroup example.newgroup moderated
Message-ID: <mvgroup-example.oldgroup-20020430@noc.example>
Approved: admin@noc.example
Control: mvgroup example.oldgroup example.newgroup moderated
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary=nxt

--nxt
Content-Type: application/news-groupinfo

For your newsgroups file:
example.newgroup        The new replacement group (Moderated)

--nxt

The moderated group example.oldgroup is replaced by
example.newgroup. Please update your configuration, and please,
if possible, arrange to file articles arriving for
example.oldgroup as if they were in example.newgroup.

--nxt
Content-Type: application/news-transmission

Newsgroups: example.admin.info
From: "example.all Administrator" <admin@noc.example>
Subject: Charter for example.newgroup
Message-ID: <mvgroup-example.newgroup-20020430@noc.example>
Distribution: local
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit

This group (formerly known as example.oldgroup) is for the
discussion of examples.
```

7.2.4. The 'checkgroups' Control Message

The "checkgroups" control message contains a list of all the valid groups in a complete hierarchy.

```
control-message     =/ Checkgroup-message
Checkgroup-message  = "checkgroups" Checkgroup-arguments
Checkgroup-arguments= [ chkscope ] [ chksernr ]
chkscope            = 1*( CFWS ["!"] newsgroup-name )
chksernr            = CFWS "#" 1*DIGIT
```

A "checkgroups" message applies to any (sub-)hierarchy with a prefix listed in the chkscope parameter, provided that the rightmost matching newsgroup-name in the list is not immediately preceded by a "!". If no chkscope parameter is given, it applies to all hierarchies for which group statements appear in the message.

> NOTE: Some existing software does not support the "chkscope" 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 do not fall within the chkscope parameter of the "checkgroups" message.

The chksernr parameter is a serial number, which can be any positive integer (e.g. 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 cannot be authenticated.

Example:

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

which includes the whole of the 'de.*' hierarchy, with the exception of its 'de.alt.*' sub-hierarchy.

The body of the message has the Content-Type "application/news-checkgroups". It asserts that the newsgroups it lists are the only newsgroups in the specified hierarchies.

> NOTE: The checkgroups message is intended to synchronize the list of newsgroups stored by a serving agent, and their newsgroup-descriptions, with the lists stored by other serving agents throughout the network. However, it might be inadvisable for the serving agent actually to create or delete any newsgroups without first obtaining the approval of its administrators for such proposed actions.

7.2.4.1. Application/news-checkgroups

The "application/news-checkgroups" body part contains a complete list of all the newsgroups in a hierarchy, their newsgroup-descriptions and their moderation status.

The MIME content type definition of "application/news-checkgroups" is:

MIME type name:	application
MIME subtype name:	news-checkgroups
Required parameters:	none
Disposition:	by default, inline
Encoding considerations:	"7bit" or "8bit" is sufficient and MUST be used to maintain compatibility.
Security considerations:	this type MUST NOT be used except as part of a checkgroups control message

The content of the "application/news-checkgroups" body part is defined as:

```
checkgroups-body    = *( valid-group CRLF )
valid-group         = newsgroups-line ; see 7.2.1.2
```

The "application/news-checkgroups" content type is used in conjunction with the "checkgroups" control message (7.2.4).

> NOTE: The possibility of removing a complete hierarchy by means of an "invalidation" line beginning with a '!' is no longer provided by this standard. The intent of the feature was widely misunderstood and it was misused more often than it was used correctly. The same effect, if required, can now be obtained by the use of an appropriate chkscope argument in conjunction with an empty checkgroups-body.

7.3. Cancel

The cancel message requests that a target article be "canceled" i.e. be withdrawn from circulation or access.

```
control-message     =/ Cancel-message
Cancel-message      = "cancel" Cancel-arguments
Cancel-arguments    = CFWS msg-id [CFWS]
```

The argument identifies the article to be cancelled by its message identifier. The body SHOULD contain an indication of why the cancellation was requested. The cancel message SHOULD be posted to the same newsgroup, with the same distribution, as the article it is attempting to cancel.

A serving agent that elects to honour a cancel message SHOULD make the article unavailable for relaying or serving (perhaps by deleting it completely). If the target article is unavailable, and the

acceptability of the cancel message cannot be established without it, activation of the cancel message SHOULD be delayed until the target article has been seen. See also sections 8.3 and 8.4.

> NOTE: It is expected that the security extension envisaged in section 7.1 will make more detailed provisions for establishing whether honouring a particular cancel message is in order. In particular, it is likely that there will be provision for the digital signature of 3rd party cancels (i.e. those issued other than by the sender, the moderator, or the injector).
>
> NOTE: A cancel submitted by the poster for an article in a moderated group will be forwarded to the moderator of that group, and it is up to that moderator to act upon it (8.7).
>
> NOTE: The former requirement [RFC 1036] that the From and/or Sender-headers of the cancel message should match those of the original article has been removed from this standard, since it only encouraged cancel issuers to conceal their true identity, and it was not usually checked or enforced by canceling software. Therefore, both the From and/or Sender-headers and any Approved-header should now relate to the entity responsible for issuing the cancel message.

7.4. Ihave, sendme

The "ihave" and "sendme" control messages implement a crude batched predecessor of the NNTP [NNTP] protocol. They are largely obsolete on the Internet, but still see use in conjunction with some transport protocols such as UUCP, especially for backup feeds that normally are active only when a primary feed path has failed. There is no requirement for relaying agents that do not support such transport protocols to implement them.

> 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:

```
control-message     =/ Ihave-message
Ihave-message       = "ihave" Ihave-arguments
Ihave-arguments     = relayer-name
control-message     =/ Sendme-message
Sendme-message      = "sendme" Sendme-arguments
Sendme-arguments    = Ihave-arguments
relayer-name        = path-identity  ; see 5.6.1
ihave-body          = *( msg-id CRLF )
sendme-body         = ihave-body
```

The body of the message consists of a list of msg-ids, one per line.

[RFC 1036] also permitted the list of msg-ids to appear in the Ihave- or Sendme-arguments with the syntax

Ihave-arguments = [FWS] *(msg-id FWS) [relayer-name]

but this form SHOULD NOT now be used, though relaying agents MAY recognize and process it for backward compatibility.

The ihave message states that the named relaying agent has received articles with the specified message identifiers, 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 identifiers to the named relaying agent.

These control messages are normally sent essentially as point-to-point messages, by using newsgroup-names in the Newsgroups-header of the form "to." followed by one (or possibly more) components in the form of a relayer-name (see section 5.5.1 which forbids "to" as the first component of a newsgroup-name). The control message SHOULD then be delivered ONLY to the relaying agent(s) identified by that relayer-name, and any relaying agent receiving such a message which includes its own relayer-name MUST NOT propagate it further. Each pair of relaying agent(s) sending and receiving these messages MUST be immediate 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 identifiers. 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.

7.5. Obsolete control messages.

The following control messages are declared obsolete by this standard:

- sendsys
- version
- whogets
- senduuname

8. Duties of Various Agents

The following section sets out the duties of various agents involved in the creation, relaying and serving of Usenet articles. Insofar as these duties are described as sequences of steps to be followed, it should be understood that it is the effect of these sequences that is important, and implementations may use any method that gives rise to

that same effect.

In this section, the word "trusted", as applied to the source of some article, means that an agent processing that article has verified, by some means, the identity of that source (which may be another agent

or a poster).

> NOTE: In many implementations, a single agent may perform various combinations of the injecting, relaying and serving functions. Its duties are then the union of the various duties concerned.

8.1. General principles to be followed

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 may 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 minutes.

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

> Cause no loops.

8.2. Duties of an Injecting Agent

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

As such, an injecting agent is considered responsible for ensuring that any article it injects conforms with the rules of this standard. It is also expected to bear some responsibility towards the rest of the network for the behaviour of its posters (and provision is therefore made for it to be easily contactable by email).

In the normal course of events, an article that has already been injected into a Netnews network will never pass through another injecting agent. So, if an injecting agent receives an otherwise valid article that has already been injected (as evidenced by the presence of an Injection-Date-header, an Injection-Info-header, or more thath one '%' path-delimiter in a Path-header) it MAY choose to reject it, but otherwise SHOULD cause it to be relayed, as it stands, by a relaying agent (8.3).

In exceptional circumstances (e.g. as part of come complex gatewaying process, or where a relaying agent considers it essential for fulfilling its responsibility towards the rest of the network) an

already injected article MAY be "reinjected" into the network. This standard does not prescribe any such circumstance; rather this is a matter of policy to be determined by the administrators of each injecting agent, who have the responsibility to ensure that no harm arises. In all other circumstances, unintented reinjection is to be avoided (see 8.8). Nevertheless, in order to preserve the integrity of the network in those special cases, this standard does set out the proper way to reinject.

8.2.1. Proto-articles

A proto-article is one that has been created by a posting agent and has not yet been injected into a Netnews system by an injecting agent. It SHOULD NOT be propagated in that form to other than injecting agents.

A proto-article has the same format as a normal article except that some of the following mandatory headers MAY be omitted: Message-Id-header, Date-header, Path-header (and even From-header if the particular injecting agent can derive that information from other sources). However, if it is intended to offer the proto-article to two or more injecting agents in parallel, then it is only the Path-header that MAY be omitted. The omitted headers MUST NOT contain invalid values; they MUST either be correct or not present at all.

> NOTE: An article that is offered for reinjection has, by definition, already been injected once, and is not therefore to be considered as a proto-article. Hence a genuine proto-article will not contain any Injection-Date-header nor any '%' path-delimiter in its Path-header. It MAY contain path-identities with other path-delimiters in the pre-injection portion of its Path-header (5.6.3).

8.2.2. Procedure to be followed by Injecting Agents

An 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. It MUST NOT forward an already injected article to a moderator.

An injecting agent processes articles as follows:

1. It MUST remove any Injection-Info- or Complaints-To-header already present (though it might be useful to copy them to suitable X-headers). It SHOULD likewise remove any NNTP-Posting-Host, X-Trace or other undocumented tracing header.

2. It SHOULD verify that the article is from a trusted source. However, it MAY allow articles in which headers contain unverified

email addresses, that is, addresses which are not known to be valid for the trusted source, and notably so if they end in ".invalid".

3. It SHOULD reject any article whose Date-header (5.1) is more than 24 hours into the future (and MAY use a margin less than that 24 hours). It MUST, except when reinjecting, reject any article with an Injection-Date-header already present (and SHOULD do likewise with any NNTP-Posting-Date-header). It MAY when reinjecting, but only if there is no Injection-Date-header present, reject any article whose Date-header appears to be stale (e.g. more than 72 hours into the past).

4. It MUST reject any article that does not have the correct mandatory headers for a proto-article (or, when reinjecting, all the mandatory headers other than Injection-Date), or which contains any header that does not have legal contents. It SHOULD reject any article which contains any header deprecated for Netnews (4.2.1).

5. If the article is rejected (for reasons given above, or for other formatting errors or matters of site policy) the posting agent SHOULD be informed (such as via an NNTP 44x response code) that posting has failed and the article MUST NOT then be processed further.

6. The Message-ID, Date and From-headers (and their contents) MUST be added when not already present (but that situation could not arise during reinjection).

7. A Path-header with a tail-entry (5.6.3) MUST be correctly added if not already present (except that it SHOULD NOT be added if the article is to be forwarded to a moderator). During reinjection, the existing Path-header SHOULD be retained.

8. The path-identity of the injecting agent with a '%' path-delimiter (5.6.2) MUST be prepended to the Path-header (which could result in more that one '%' path-delimiter in the case of reinjection); moreover, that path-identity MUST be an FQDN mailable address (5.6.2).

9. An Injection-Info-header (6.19) SHOULD be added, identifying the trusted source of the article, and a suitable Complaints-To-header (6.20) MAY be added (except that these two headers MUST NOT be added if the article is to be forwarded to a moderator).

10. The injecting agent MUST NOT alter the body of the article in any way. It MAY, except when reinjecting, add other headers not already provided by the poster, but SHOULD NOT alter, delete, or reorder any existing header, with the specific exception of "tracing" headers such as Injection-Info and Complaints-To, which are to be removed as already mentioned.

11. If the Newsgroups-header contains no moderated groups, or if it contains an Approved-header, the injecting agent MUST then add an Injection-Date-header (5.7) if one is not already present, but it MUST NOT alter, or remove, an already present Injection-Date-header (and likewise SHOULD NOT alter, or remove, an already

present NNTP-Posting-Date-header). Finally, it forwards the article to one or more relaying or serving agents.

12. Otherwise, when the Newsgroups-header contains one or more moderated groups and the article does NOT contain an Approved-header, the injecting agent MUST forward it to the moderator of the first (leftmost) moderated group listed in the Newsgroups-header via email. There are two possibilities for doing this:

    (a) The complete article is encapsulated (headers and all) within the email, preferably using the Content-Type "application/news-transmission" (6.21.4.1) with any usage parameter set to "moderate". Moreover, there SHOULD NOT be more than one encapsulated article within the one email. This method has the advantage of removing any possible conflict between Netnews and Email headers, or of changes to those headers during transport through email.

    (b) The article is sent as an email as it stands, with the addition of such extra headers (e.g. a To-header) as are necessary for an email.

    Although both of these methods have seen use in the past, the preponderance of current usage on Usenet has been for method (b) and many moderators are ill-prepared to deal with method (a). Therefore, method (a) SHOULD NOT be used until such time as the majority of moderators are able to accept it.

13. This standard does not prescribe how the email address of the moderator is to be determined, that being a matter of policy to be arranged by the agency responsible for the oversight of each hierarchy. Nevertheless, there do exist various agents worldwide which provide the service of forwarding to moderators, and the address to use with them is obtained as follows:

    (a) Each '.' in the newsgroup-name is replaced with a '-'.

    (b) The result of these operations is used as the local-part of the mailbox of the agent. For example, articles intended for "news.announce.important" would be emailed to "news-announce-important@forwardingagent.example".

8.3. Duties of a Relaying Agent

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

A relaying agent processes articles as follows:

1. It MUST verify the leftmost entry in the Path-header and then prepend its own path-identity with a '/' path-delimiter, and possibly also the verified path-identity of its source with a '?'

path-delimiter (5.6.2).

2. It MUST examine the Injection-Date-header (or, if that is absent, the Date-header) and reject the article as stale (5.7) if that predates the earliest articles of which it normally keeps record, or if it is more than 24 hours into the future (the margin MAY be less than that 24 hours).

3. It MUST reject any article that does not have the correct mandatory headers (section 5) present with legal contents.

4. It SHOULD reject any article whose optional headers (section 6) do not have legal contents.

5. It SHOULD reject any article that has already been sent to it (a database of message identifiers of recent messages is usually kept and matched against).

6. It SHOULD reject any article that matches an already received cancel message (or an equivalent Supersedes-header) issued by its poster or by some other trusted entity.

7. It MAY reject any article without an Approved-header posted to newsgroups known to be moderated (this practice is strongly recommended, but the information necessary to do it may not be available to all agents).

8. It then passes articles which match mutually agreed criteria on to neighbouring relaying and serving agents. However, it SHOULD NOT forward articles to sites whose path-identity is already in the Path-header.

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

If the article is rejected as being invalid, unwanted or unacceptable due to site policy, the agent that passed the article to the relaying agent SHOULD be informed (such as via an NNTP 43x response code) that relaying failed. In order to prevent a large number of error messages being sent to one location, relaying agents MUST NOT inform any other external entity that an article was not relayed UNLESS that external entity has explicitly requested that it be informed of such errors.

   NOTE: In order to prevent overloading, relaying agents should not routinely query an external entity (such as a DNS-server) in order to verify an article (though a local cache of the required information might usefully be consulted).

Relaying agents MUST NOT alter, delete or rearrange any part of an

article expect for headers designated as variant (4.2.5.3).

8.4. 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 articles in each newsgroup identified by an article-locator (usually in the form of a decimal number - see 6.16).

> NOTE: Since control messages are often of interest, but should not be displayed as normal articles in regular newsgroups, it is common for serving agents to make them available in a pseudo-newsgroup named "control" or in a pseudo-newsgroup in a sub-hierarchy under "control." (e.g. "control.cancel").

A serving agent processes articles as follows:

1. It MUST verify the leftmost entry in the Path-header and then prepend its own path-identity with a '/' path-delimiter, and possibly also the verified path-identity of its source with a '?' path-delimiter (5.6.2).

2. It MUST examine the Injection-Date-header (or, if that is absent, the Date-header) and reject the article as stale (5.7) if that predates the earliest articles of which it normally keeps record, or if it is more than 24 hours into the future (the margin MAY be less than that 24 hours).

3. It MUST reject any article that does not have the correct mandatory headers (section 5) present, or which contains any header that does not have legal contents.

4. It SHOULD reject any article that has already been sent to it (a database of message identifiers of recent messages is usually kept and matched against).

5. It SHOULD reject any article that matches an already received cancel message (or an equivalent Supersedes-header) issued by its poster or by some other trusted entity.

6. It MUST reject any article without an Approved-header posted to any moderated newsgroup which it is configured to receive, and it MAY reject such articles for any newsgroup it knows to be moderated.

7. It MUST remove any Xref-header (6.16) from each article. It then MAY (and usually will) generate a fresh Xref-header.

8. Finally, it stores the article in its news database.

8.5. 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 according to this standard and other applicable policies. In particular, they MUST NOT create any Injection-Date-, Injection-Info- or Complaints-To-header.

Posting agents meant for use by ordinary posters SHOULD reject any attempt to post an article which cancels or Supersedes another article of which the poster is not the author.

8.6. 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. Followup agents MUST create valid followups and are subject to special requirements involving certain inheritable (4.2.5.2) and other headers. Wherever in the following it is stated that, "by default", a header is to be taken from some header in the precursor, it means that its initial content (plus its extension-parameters, if any) are to be copied from those of that precursor header. However, posters MAY then override that default before posting if they so wish.

> NOTE: There is no provision in this standard for a followup to have more than one precursor (though it might be permitted in some future extension).

1. The Newsgroups-header (5.5) SHOULD by default be taken from the precursor's Followup-To-header if present, and otherwise from the precursor's Newsgroups-header. However, if the content of that Followup-To-header consists of "poster" (and the user does not override it), then the followup MUST NOT be posted but, rather, is to be emailed to the precursor's poster.

2. The Subject-header SHOULD by default be taken from that of the precursor's Subject-header. The case sensitive string "Re: " (known as a "back reference") MAY be prepended to its Subject-Content unless it already begins with that string.

> NOTE: The practice of prepending such a "Re: " (which is an abbreviation for the Latin "In re", meaning "in the matter of", and not an abbreviation of "Reference" as is sometimes erroneously supposed) is widespread, but reading agents cannot rely on all followup Subject-headers using it, nor can it be assumed that reading agents will understand any back-reference other than a single "Re: ".

[Shmuel wants something like "... nor that the "Re:" was even intended as a back-reference."]

> Some reading agents, notably those which choose to use the Subject-header to enable a simple form of thread sorting, need

to be able to recognize the presence of a back-reference.

[Various other snippets of text have been suggested for the NOTE, including the following:

NOTE: Some reading agents expect a single "Re: " at the beginning of the Subject-content of a followup, and consider this when sorting articles for display. This is unreliable, and an intrusion on the unstructured nature of the header, but widespread.

NOTE: Some reading agents expect this string at the beginning of the Subject-content of a followup, and consider this when sorting articles for display. Further, most expect just one instance of "Re: ", and do not treat any other string in this manner.

NOTE: Further discussion of this (user interface) issue can be found in [USEAGE] section x.x.

Although "Re: " is case-sensitive, there has historically been some disagreement about that, so it is best to check for its presence with a case-insensitive test.

End of snippets.]

3. The Distribution-header (6.6) SHOULD by default be taken from the precursor's Distribution-header, if any.

4. If the precursor did not have a References-header (6.10), the followup's References-content MUST be formed by the message identifier of the precursor. A followup to an article which had a References-header MUST have a References-header containing the precursor's References-content (subject to trimming as described below) plus the precursor's message identifier appended to the end of the list (separated from it by CFWS).

   Followup agents MAY trim References-headers which have grown to excessive length, but the first and last message identifiers from the precursor MUST NOT be removed.

5. If the precursor contains a Mail-Copies-To-header (6.8), the actions to be taken, in accordance with the Mail-Copies-To-content, (and subject to manual override by the poster) are as follows:

   "nobody" (or when the header is absent)
   The followup agent SHOULD NOT email a copy of a posted followup to the poster of the precursor.

   "poster"
   The followup agent SHOULD (if it has the necessary capability) email a copy of a posted followup, which MUST then be sent to the address(es) in the Reply-To-header, and in the absence of that to the address(es) in the From-header.

a copy-addr
: The followup agent SHOULD likewise email a copy of a posted followup, which MUST then be sent to the copy-addr.

When emailing a copy, the followup agent SHOULD also include a "Posted-And-Mailed: yes" header (6.9).

Followup agents SHOULD NOT attempt to send email to any address ending in ".invalid".

8.7. Duties of a Moderator

A Moderator receives news articles by email, decides whether to accept them and, if so, either injects them into the news stream or forwards them to further moderators.

Articles will be received by the moderator either encapsulated as an object of Content-Type application/news-transmission (or possibly encapsulated but without an explicit Content-Type-header), or else directly as an email already containing all the headers appropriate for a Netnews article (see 8.2.2). Moderators SHOULD be prepared to accept articles in either format.

A moderator processes an article, as submitted to any newsgroup that he moderates, as follows:

1. He decides, on the basis of whatever moderation policy applies to his group, whether to accept or reject the article. He MAY do this manually, or else partially or wholly with the aid of appropriate software for whose operation he is then responsible. If the article is a cancel nessage (7.3) issued by the poster of an earlier article, then he is expected to cancel that earlier article (in which case there is no more to be done). He MAY modify the article if that is in accordance with the applicable moderation policy (and in particular he MAY remove redundant headers and add Comments and other informational headers). He also needs to be aware if any change he makes to the article will invalidate some authentication check provided by the poster or by an earlier moderator.

   If the article is rejected, then it normally fails for all the newsgroups for which it was intended. If it is accepted, the moderator proceeds with the following steps.

2. If the Newsgroups-header contains further moderated newsgroups for which approval has not already been given, he adds an indication (identifying both himself and the name of the group) that he approves the article, and then forwards it to the moderator of the leftmost unapproved group (which, if this standard has been followed correctly, will generally be the next moderated group to the right of his own). There are two ways to do this:

   (a) He emails it to the submission address of the next moderator (see section 8.2.2 for the proper method of doing this), or

(b) he rotates the newsgroup-names in the Newsgroups-header to the left so that the targeted group is the leftmost moderated group in that header, and injects it as below (thus causing

the injecting agent to email it to the correct moderator). However, he MUST first ensure that the article contains no Approved-header.

NOTE: This standard does not prescribe how a moderator's approval is to be indicated (though a future standard may do so). Possible methods include adding an Approved header (or a similar but differently named header if method (b) is being used) listing all the approvals made so far, or adding a separate header for each individual approval (the header X-Auth is sometimes used for this purpose). The approval may also be confirmed with some form of digital signature (7.1).

3. If the Newsgroups-header contains no further unapproved moderated groups, he adds an Approved-header (6.14) identifying himself and, insofar as is possible, all the other moderators who have approved the article. He thus assumes responsibility for having ensured that the article was acceptable to the moderators of all the moderated groups involved.

4. The Date-header SHOULD be retained. Any Injection-Date-header already present (though there should be none) MUST be removed. Exceptionally, if it is known that the injecting agent does not yet support the Injection-Date-header and the Date-header appears to be stale (5.7) for reasons understood by the moderator (e.g. delays in the moderation process) he MAY substitute the current date. The Message-ID-header SHOULD also be retained unless it is obviously non-compliant with this standard.

   NOTE: A message identifier created by a conforming posting or injecting agent, or even by a mail user agent conforming to [RFC 2822], may reasonably be supposed to be conformant (and will, in any case, be caught by the injecting agent if it is not).

5. Any variant headers (4.2.5.3) MUST be removed, except that a Path-header MAY be truncated to only its pre-injection region (5.6.3). Any Injection-Date-header (5.7), Injection-Info-header (6.19) or Complaints-To-header (6.20) MUST be removed (though in fact there should be none of these).

6. He then causes the article to be injected, having first observed all the duties of a posting agent.

   NOTE: This standard does not prescribe how the moderator or moderation policy for each newsgroup is established; rather it assumes that whatever agencies are responsible for the relevant network or hierarchy (1.1) will have made appropriate arrangements in that regard.

8.8. Duties of a Gateway

A Gateway transforms an article into the native message format of another medium, or translates the messages of another medium into news articles. Encapsulation of a news article into a message of MIME

type application/news-transmission, or the subsequent undoing of that encapsulation, is not gatewaying, since it involves no transformation of the article.

There are two basic types of gateway, the Outgoing Gateway that transforms a news article into a different type of message, and the Incoming Gateway that transforms a message from another medium into a news article and injects it into a news system. These are handled separately below.

The primary dictat for a gateway is:

> Above all, prevent loops.

Transformation of an article into another medium stands a very high chance of discarding or interfering with the protection inherent in the news system against duplicate articles. The most common problem caused by gateways is "spews," gateway loops that cause previously posted articles to be reinjected repeatedly into Usenet. To prevent this, a gateway MUST take precautions against loops, as detailed below.

If bidirectional gatewaying (both an incoming and an outgoing gateway) is being set up between Netnews and some other medium, the incoming and outgoing gateways SHOULD be coordinated to avoid unintended reinjection of gated articles. Circular gatewaying (gatewaying a message into another medium and then back into Netnews) SHOULD NOT be done; encapsulation of the article SHOULD be used instead where this is necessary.

A second general principal of gatewaying is that the transformations applied to the message SHOULD be as minimal as possible while still accomplishing the gatewaying. Every change made by a gateway potentially breaks a property of one of the media or loses information, and therefore only those transformations made necessary by the differences between the media should be applied.

It is worth noting that safe bidirectional gatewaying between a mailing list and a newsgroup is far easier if the newsgroup is moderated. Posts to the moderated group and submissions to the mailing list can then go through a single point that does the necessary gatewaying and then sends the message out to both the newsgroup and the mailing list at the same time, eliminating most of the possibility of loops. Bidirectional gatewaying between a mailing list and an unmoderated newsgroup, in contrast, is difficult to do correctly and is far more fragile.

Newsgroups intended to be bidirectionally gated to a mailing list SHOULD therefore be moderated where possible, even if the moderator

is a simple gateway and injecting agent that correctly handles crossposting to other moderated groups and otherwise passes all traffic.

8.8.1. Duties of an Outgoing Gateway

From the perspective of Netnews, an outgoing gateway is just a special type of reading agent. The exact nature of what the outgoing gateway will need to do to articles depends on the medium to which the articles are being gated. The operation of the outgoing gateway is only subject to additional constraints in the presence of one or more corresponding incoming gateways back from that medium to Netnews, since this opens the possibility of loops.

In general, the following practices are recommended for all outgoing gateways, regardless of whether there is known to be a related incoming gateway, both as a precautionary measure and as a guideline to quality of implementation.

1. The message identifier of the news article should be preserved if at all possible, preferably as or within the corresponding unique identifier of the other medium, but if not at least as a comment in the message. This helps greatly with preventing loops.

2. The Date and Injection-Date of the news article should also be preserved if possible, for similar reasons.

3. The message should be tagged in some way so as to prevent its reinjection into Netnews. This may be impossible to do without knowledge of potential incoming gateways, but it is better to try to provide some indication even if not successful; at the least, a human-readable indication that the article should not be gated back to Netnews can help locate a human problem.

4. Netnews control messages should not be gated to another medium unless they would somehow be meaningful in that medium.

8.8.2. Duties of an Incoming Gateway

The incoming gateway has the serious responsibility of ensuring that all of the requirements of this standard are met by the articles that it forms. In addition to its special duties as a gateway, it bears all of the duties and responsibilities of an injecting agent as well, and additionally has the same responsibility of a relaying agent to reject articles that it has already gatewayed.

An incoming gateway MUST NOT gate the same message twice. It may not be possible to ensure this in the face of mangling or modification of the message, but at the very least a gateway, when given a copy of a message it has already gated identical except for trace headers (like Received in Email or Path in Netnews) MUST NOT gate the message again. An incoming gateway SHOULD take precautions against having this rule bypassed by modifications of the message that can be anticipated.

News articles prepared by gateways MUST be legal news articles. In particular, they MUST include all of the mandatory headers, MUST fully conform to the restrictions on said headers, and SHOULD exclude

any deprecated headers (4.2.1). This often requires that a gateway function not only as a relaying agent, but also partly as a posting agent, aiding in the synthesis of a conforming article from non-conforming input.

Incoming gateways MUST NOT pass control messages (articles containing a Control- or Supersedes-header) without removing or renaming that header. Gateways MAY, however, generate their own cancel messages, under the general allowance for injecting agents to cancel their own messages (7.3). If a gateway receives a message that it can determine is a valid equivalent of a cancel message in the medium it is gatewaying, it SHOULD discard that message without gatewaying it, generate a corresponding cancel message of its own, and inject that cancel message.

Incoming gateways MUST NOT inject control messages other than cancels. Encapsulation SHOULD be used instead of gatewaying, when direct posting is not possible or desirable.

> NOTE: It is not unheard of for mail-to-news gateways to be used to post control messages, but encapsulation should be used for these cases instead. Gateways by their very nature are particularly prone to loops. Spews of normal articles are bad enough; spews of control messages with special significance to the news system, possibly resulting in high processing load or even email sent for every message received, are catastrophic. It is far preferable to construct a system specifically for posting control messages that can do appropriate consistency checks and authentication of the originator of the message.

If there is a message identifier that fills a role similar to that of the Message-ID-header in news, it SHOULD be used in the formation of the message identifier of the news article, perhaps with transformations required to meet the uniqueness requirement of Netnews and with the removal of any comments so as to comply with the syntax in section 5.3. Such transformations SHOULD be designed so that two messages with the same identifier generate the same Message-ID-header.

> NOTE: Message identifiers play a central role in the prevention of duplicates, and their correct use by gateways will do much to prevent loops. Netnews does, however, require that message identifiers be unique, and therefore message identifiers from other media may not be suitable for use without modification. A balance must be struck by the gateway between preserving information used to prevent loops and generating unique message identifiers.

Exceptionally, if there are multiple incoming gateways for a

particular set of messages, each to a different newsgroup(s), each one SHOULD generate a message identifier unique to that gateway. Each incoming gateway nonetheless MUST ensure that it does not gate the same message twice.

NOTE: Consider the example of two gateways of a given mailing list into the world-wide Usenet newsgroups, both of which preserve the email message identifier. Each newsgroup may then receive a portion of the messages (different sites seeing different portions). In these cases, where there is no one "official" gateway, some other method of generating message identifiers has to be used to avoid collisions. It would obviously be preferable for there to be only one gateway which crossposts, but this may not be possible to coordinate.

If no date information is available, the gateway MAY supply a Date-header with the gateway's current date. If no injection-date information is available, the gateway MUST supply an Injection-Date-header with whatever date information is available, and otherwise with the gateway's current date. If only partial information is available (e.g. date but not time), this SHOULD be fleshed out to a full Date- and/or Injection-Date-header by adding default values rather than discarding this information. Only in very exceptional circumstances should Date information be discarded, as it plays an important role in preventing reinjection of old messages.

An incoming gateway MUST add a Sender-header to the news article it forms containing the mailbox of the administrator of the gateway. Problems with the gateway may be reported to this mailbox. The display-name portion of this mailbox SHOULD indicate that the entity responsible for injection of the message is a gateway. If the original message already had a Sender-header, it SHOULD be renamed so that its contents can be preserved.

8.8.3. Example

To illustrate the type of precautions that should be taken against loops, here is an example of the measures taken by one particular combination of mail-to-news and news-to-mail gateways at Stanford University designed to handle bidirectional gatewaying between mailing lists and unmoderated groups.

1. The news-to-mail gateway preserves the message identifier of the news article in the generated email message. The mail-to-news gateway likewise preserves the email message identifier provided that it is syntactically valid for Netnews. This allows the news system's built-in suppression of duplicates to serve as the first line of defense against loops.

2. The news-to-mail gateway adds an X-Gateway-header to all messages it generates. The mail-to-news gateway discards any incoming messages containing this header. This is robust against mailing list managers that replace the message identifier, and against any number of email hops, provided that the other message headers are

preserved.

3. The mail-to-news gateway inserts the host name from which it received the email message in the pre-injection region of the Path (5.6.3). The news-to-mail gateway refuses to gateway any message

that contains the list server name in the pre-injection region of its Path-header. This is robust against any amount of munging of the message headers by the mailing list, provided that the email only goes through one hop.

4. The mail-to-news gateway is designed never to generate bounces to the envelope sender. Instead, articles that are rejected by the news server (for reasons not warranting silent discarding of the message) result in a bounce message sent to an errors address known not to forward to any mailing lists, so that they can be handled by the news administrators.

These precautions have proven effective in practice at preventing loops for this particular application (bidirectional gatewaying between mailing lists and locally distributed newsgroups where both gateways can be designed together). General gatewaying to world-wide newsgroups poses additional difficulties; one must be very wary of strange configurations, such as a newsgroup gated to a mailing list which is in turn gated to a different newsgroup.

9. Security and Related Considerations

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.

9.1. Leakage

Articles which are intended to have restricted distribution are dependent on the goodwill of every site receiving them. The "Archive: no" header (6.12) is available as a signal to automated archivers not to file an article, but that cannot be guaranteed.

The Distribution-header makes provision 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.

The flooding algorithm is extremely good at finding any path by which articles can leave a subnet with supposedly restrictive boundaries, and substantial administrative effort is required to avoid this. Organizations wishing to control such leakage are strongly advised to designate a small number of official gateways to handle all news exchange with the outside world (however, making such gateways too restrictive can also encourage the setting up of unofficial paths which can be exceedingly hard to track down).

The sendme control message (7.4), insofar as it is still used, can be used to request articles with a given message identifier, even one that is not supposed to be supplied to the requestor.

9.2. Attacks

9.2.1. Denial of Service

The proper functioning of individual newsgroups can be disrupted by the massive posting of "noise" articles, by the repeated posting of identical or near identical articles, by posting followups unrelated to their precursors, or which quote their precursors in full with the addition of minimal extra material (especially if this process is iterated), and by crossposting to, or setting followups to, totally unrelated newsgroups.

Many have argued that "spam", massively multiposted (and to a lesser extent massively crossposted) articles, usually for advertising purposes, also constitutes a DoS attack in its own regard. This may be so.

Such articles intended to deny service, or other articles of an inflammatory nature, may also have their From or Reply-To addresses set to valid but incorrect email addresses, thus causing large volumes of email to descend on the true owners of those addresses.

Similar effects could be caused by any email header which could cause every reading agent receiving it to take some externally visible action. For example, the Disposition-Notification-To-header defined in [RFC 2298] could cause huge numbers of acknowledgements to be emailed to an unsuspecting third party (for which reason [RFC 2298] declares that that header SHOULD NOT be used in Netnews).

It is a violation of this standard for a poster to use as his address a mailbox which he is not entitled to use. Even addresses with an invalid local-part but a valid domain can cause disruption to the administrators of such domains. Posters who wish to remain anonymous or to prevent automated harvesting of their addresses, but who do not care to take the additional precautions of using more sophisticated anonymity measures, should avoid that violation by the use of addresses ending in the ".invalid" top-level-domain (see 5.2).

A malicious poster may also prevent his article being seen at a particular site by preloading that site into the Path-header (5.6.1) and may thus prevent the true owner of a forged From or Reply-To address from ever seeing it.

A malicious complainer may submit a modified copy of an article (e.g. with an altered Injection-Info-header) to the administrator of an injecting agent in an attempt to discredit the author of that article and even to have his posting privileges removed. Administrators should therefore obtain a genuine copy of the article from their own serving agent before taking such precipitate action.

Administrative agencies with responsibility for establishing policies in particular hierarchies can and should set bounds upon the behaviour that is considered acceptable within those hierarchies (for example by promulgating charters for individual newsgroups, and other

codes of conduct).

Whilst this standard places an onus upon injecting agents to bear responsibility for the misdemeanours of their posters (which includes non-adherence to established policies of the relevant hierarchies as provided in section 8.2), and to provide assistance to the rest of the network by making proper use of the Injection-Info- (6.19) and Complaints-To- (6.20) headers, it makes no provision for enforcement, which may in consequence be patchy. Nevertheless, injecting sites which persistently fail to honour their responsibilities or to comply with generally accepted standards of behaviour are likely to find themselves blacklisted, with their articles refused propagation and even subject to cancellation, and other relaying sites would be well advised to withdraw peering arrangements from them.

9.2.2. Compromise of System Integrity

The posting of unauthorized (as determined by the policies of the relevant hierarchy) control messages can cause unwanted newsgroups to be created, or wanted ones removed, from serving agents. Administrators of such agents SHOULD therefore take steps to verify the authenticity of such control messages, either by manual inspection (particularly of the Approved-header) or by checking any digital signatures that may be provided (see 7.1). In addition, they SHOULD periodically compare the newsgroups carried against any regularly issued checkgroups messages, or against lists maintained by trusted servers and accessed by out-of-band protocols such as FTP or HTTP.

Malicious cancel messages (7.3) can cause valid articles to be removed from serving agents. Administrators of such agents SHOULD therefore take steps to verify that they originated from the (apparent) poster, the injector or the moderator of the article, or that in other cases they came from a place that is trusted to work within established policies and customs. Such steps SHOULD include the checking of any digital signatures, or other security devices, that may be provided (see 7.1). Articles containing Supersedes-headers (6.15) are effectively cancel messages, and SHOULD be subject to the same checks. Currently, many sites choose to ignore all cancel messages on account of the difficulty of conducting such checks.

Improperly configured serving agents can allow articles posted to moderated groups onto the net without first being approved by the moderator. Injecting agents SHOULD verify that moderated articles were received from one of the entities given in their Approved-headers and/or check any digital signatures that may be provided (see 7.1).

The filename parameter of the Archive-header (6.12) can be used to attempt to store archived articles in inappropriate locations. Archiving sites should be suspicious of absolute filename parameters, as opposed to those relative to some location of the archiver's choosing.

There may be weaknesses in particular implementations that are subject to malicious exploitation. In particular, it has not been unknown for complete shell scripts to be included within Control-headers. Implementors need to be aware of this.

Reading agents should be chary of acting automatically upon MIME objects with an "application" Content-Type that could change the state of that agent, except in contexts where such applications are specifically expected (see 6.21). Even the Content-Type "text/html" could have unexpected side effects on account of embedded objects, especially embedded executable code or URLs that invoke non-news protocols such as HTTP [RFC 2616]. It is therefore generally recommended that reading agents do not enable the execution of such code (since it is extremely unlikely to have a valid application within Netnews) and that they only honour URLs referring to other parts of the same article.

Non-printable characters embedded in article bodies may have surprising effects on printers or terminals, notably by reconfiguring them in undesirable ways which may become apparent only after the reading agent has terminated.

9.3. Liability

There is a presumption that a poster who sends an article to Usenet intends it to be stored on a multitude of serving agents, and has therefore given permission for it to be copied to that extent. Nevertheless, Usenet is not exempt from the Copyright laws, and it should not be assumed that permission has been given for the article to be copied outside of Usenet, nor for its permanent archiving contrary to any Archive-header that may be present.

Posters also need to be aware that they are responsible if they breach Copyright, Libel, Harassment or other restrictions relating to material that they post, and that they may possibly find themselves liable for such breaches in jurisdictions far from their own. Serving agents may also be liable in some jurisdictions, especially if the breach has been explicitly drawn to their attention.

Users who are concerned about such matters should seek advice from competent legal authorities.

10. IANA Considerations

10.1. Media Types

IANA is requested to register the following media types, described elsewhere in this standard for use with the Content-Type-header, in the IETF tree in accordance with the procedures set out in [RFC 2048].

```
      application/news-transmission  (6.21.4.1)
      application/news-groupinfo     (7.2.1.2)
      application/news-checkgroups   (7.2.4.1)
```

IANA is also requested to change the status of the following media type to "OBSOLETE".

message/news (6.21.4.2)

NOTE: "Application/news-transmission" is an update, with clarification and additional optional parameters, to an existing registration. "Message/rfc822" should now be used in place of the obsoleted "message/news".

10.2. Header Field Registry

IANA is requested to register the following headers in the Permanent Message Header Field Repository, in accordance with the procedures set out in [KLYNE].

Header field name	Applicable protocol	Status	Author/Change controller	Specification document
Date	netnews	standard	IETF	[USEFOR] 5.1
From	netnews	standard	IETF	[USEFOR] 5.2
Message-ID	netnews	standard	IETF	[USEFOR] 5.3
Subject	netnews	standard	IETF	[USEFOR] 5.4
Newsgroups	netnews	standard	IETF	[USEFOR] 5.5
Newsgroups	mail	standard	IETF	[USEFOR] 5.5
Path	netnews	standard	IETF	[USEFOR] 5.6
Reply-To	netnews	standard	IETF	[USEFOR] 6.1
Sender	netnews	standard	IETF	[USEFOR] 6.2
Organization	netnews	standard	IETF	[USEFOR] 6.3
Keywords	netnews	standard	IETF	[USEFOR] 6.4
Summary	netnews	standard	IETF	[USEFOR] 6.5
Distribution	netnews	standard	IETF	[USEFOR] 6.6
Followup-To	netnews	standard	IETF	[USEFOR] 6.7
Mail-Copies-To	netnews	standard	IETF	[USEFOR] 6.8
Posted-And-Mailed	netnews	standard	IETF	[USEFOR] 6.9
References	netnews	standard	IETF	[USEFOR] 6.10
Expires	netnews	standard	IETF	[USEFOR] 6.11
Archive	netnews	standard	IETF	[USEFOR] 6.12
Control	netnews	standard	IETF	[USEFOR] 6.13
Approved	netnews	standard	IETF	[USEFOR] 6.14
Supersedes	netnews	standard	IETF	[USEFOR] 6.15
Xref	netnews	standard	IETF	[USEFOR] 6.16
Lines	netnews	standard	IETF	[USEFOR] 6.17
User-Agent	netnews	standard	IETF	[USEFOR] 6.18
User-Agent	mail	standard	IETF	[USEFOR] 6.18
Injection-Info	netnews	standard	IETF	[USEFOR] 6.19
NNTP-Posting-Host	netnews	deprecated	IETF	[USEFOR] 6.19
NNTP-Posting-Date	netnews	deprecated	IETF	[USEFOR] 5.7
Complaints-To	netnews	standard	IETF	[USEFOR] 6.20

Also-Control	netnews	obsoleted	IETF	[USEFOR] 6.22
See-Also	netnews	obsoleted	IETF	[USEFOR] 6.22
Article-Names	netnews	obsoleted	IETF	[USEFOR] 6.22
Article-Updates	netnews	obsoleted	IETF	[USEFOR] 6.22

11. References

[ANSI X3.4] "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 3166] "Codes for the representation of names of countries and their subdivisions -- Part 1: Country codes", ISO 3166, 1997.

[ISO/IEC 10646] "International Standard - Information technology - Universal Multiple-Octet Coded Character Set (UCS) - Part 1: Architecture and Basic Multilingual Plane", ISO/IEC 10646-1:2000, 2000.

[KLYNE] G. Klyne, M. Nottingham, and J. Mogul, "Registration procedures for message header fields", draft-klyne-msghdr-registry-07.txt.

[NNTP] S. Barber, "Network News Transport Protocol", draft-ietf-nntpext-base-*.txt.

[PGPVERIFY] David Lawrence, <ftp://ftp.isc.org/pub/pgpcontrol/README.html>.

[RFC 1034] P. Mockapetris, "Domain Names - Concepts and Facilities", RFC 1034, November 1987.

[RFC 1036] M. Horton and R. Adams, "Standard for Interchange of USENET Messages", RFC 1036, December 1987.

[RFC 1864] J. Myers and M. Rose, "The Content-MD5 Header Field", RFC 1864, October 1995.

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

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

[RFC 2047] K. Moore, "MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text", RFC 2047, November 1996.

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

[RFC 2119] S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, March 1997.

[RFC 2142] D. Crocker, "Mailbox Names for Common Services, Roles and

Functions", RFC 2142, May 1997.

[RFC 2156] S. Kille, "MIXER (Mime Internet X.400 Enhanced Relay): Mapping between X.400 and RFC 822/MIME", RFC 2156, January 1998.

[RFC 2183] R. Troost, S. Dorner, and K.Moore, "Communicating Presentation Information in Internet Messages: The Content-Disposition Header Field", RFC 2183, August 1997.

[RFC 2231] N. Freed and K. Moore, "MIME Parameter Value and Encoded Word Extensions: Character Sets, Languages, and Continuations", RFC 2231, November 1997.

[RFC 2234] D. Crocker and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.

[RFC 2298] R. Fajman, "An Extensible Message Format for Message Disposition Notifications", RFC 2298, March 1998.

[RFC 2373] R. Hinden and S. Deering, "IP Version 6 Addressing Architecture", RFC 2373, July 1998.

[RFC 2557] J. Palme, A. Hopmann, and N. Shelness, "MIME Encapsulation of Aggregate Documents, such as HTML (MHTML)", RFC 2557, March 1999.

[RFC 2606] D. Eastlake and A. Panitz, "Reserved Top Level DNS Names", RFC 2606, June 1999.

[RFC 2616] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[RFC 2822] P. Resnick, "Internet Message Format", RFC 2822, April 2001.

[RFC 2978] N. Freed and J. Postel, "IANA Charset Registration Procedures", RFC 2978, October 2000.

[RFC 3066] H. Alvestrand, "Tags for the Identification of Languages", RFC 3066, January 2001.

[RFC 3156] M. Elkins, D. Del Torto, R. Levien, and T. Roessler, "MIME Security with OpenPGP", RFC 3156, August 2001.

[RFC 3282] H. Alvestrand, "Content Language Headers", RFC 3282, May 2002.

[RFC 822] D. Crocker, "Standard for the Format of ARPA Internet Text Messages.", STD 11, RFC 822, August 1982.

[RFC 850] Mark R. Horton, "Standard for interchange of Usenet
messages", RFC 850, June 1983.

[RFC 976] Mark R. Horton, "UUCP mail interchange format standard", RFC 976, February 1986.

[RFC Errata] RFC Editor, "RFC Errata", <http"//www/rfc-editor.org/errata.html>.

[Son-of-1036] Henry Spencer, "News article format and transmission", <ftp://ftp.zoo.toronto.edu/pub/news.txt.Z>, June 1994.

[UNICODE 3.0] The Unicode Consortium, "The Unicode Standard - Version 3.0", Addison-Wesley, 2000.

[UNICODE 3.1] The Unicode Consortium, "The Unicode Standard - Version 3.1, being an amendment to [UNICODE 3.0]", Unicode Standard Annex #27 <http://www.unicode.org/unicode/reports/tr27>, 2001.

[UNICODE 3.2] The Unicode Consortium, "The Unicode Standard - Version 3.2, being an amendment to [UNICODE 3.1]", Unicode Standard Annex #28 <http://www.unicode.org/unicode/reports/tr28>, 2002.

[USEAGE] Draft in preparation.

[USEFOR] This Standard.

12. Acknowledgements

The editor wishes to thank the following members of the IETF Usenet Format Working Group who made significant contributions to this endeavour (however, inclusion in this list does not imply that a person approves of everything contained herein).

Per Abrahamsen	Brian Kelly
Peter Alfredsen	Evan Kirshenbaum
Russ Allbery	Brad Knowles
Greg Andruk	Kent Landfield
Ralph Babel	David C. Lawrence
Stan Barber	Bruce Lilly
Dave Barr	Simon Lyall
Ian Bell	Todd Michel McComb
G. James Berigan	Denis McKeon
Terje Bless	Seymour J. Metz
Seth Breidbart	John Moreno
Buddha Buck	Chris Newman
Forrest J. Cavalier III	Dirk Nimmich
Evan Champion	Paul Overell
Maurizio Codogno	Jacob Palme
Don Croyle	Brian Palmer
Matt Curtin	Pete Resnick
Bill Davidsen	Jon Ribbens

Ian Davis	Dan Ritter
Jean-Marc Desperrier	Thomas Roessler
Martin J. Duerst	Doug Royer
Claus Andre Faerber	Frederic Senis

Clive D.W. Feather	Erland Sommarskog
David Formosa	Henry Spencer
Marty Fouts	John Stanley
Benjamin Franz	Brad Templeton
Andrew Gierth	Florian Weimer
Jonathan Grobe	Curt Welch
Thomas Gschwind	Curtis Whalen
Kai Henningsen	Leonid Yegoshin
Lars Magne Ingebrigtsen	Jamie Zawinski

13. Contact Address

Editor

Charles. H. Lindsey
5 Clerewood Avenue
Heald Green
Cheadle
Cheshire SK8 3JU
United Kingdom
Phone: +44 161 436 6131
Email: chl@clw.cs.man.ac.uk

[

Working group chairs

Andrew Gierth <andrew@erlenstar.demon.co.uk>
Pete Resnick <presnick@qualcomm.com>

]

Comments on this draft should preferably be sent to the mailing list of the Usenet Format Working Group at

usenet-format@landfield.com.

This draft expires six months after the date of publication (see Page 1) (i.e. in November 2004).

Appendix A.1 - A-News Article Format

The obsolete "A News" article format consisted of exactly five lines of header information, followed by the body. For example:

```
Aeagle.642
news.misc
cbosgd!mhuxj!mhuxt!eagle!jerry
Fri Nov 19 16:14:55 1982
Usenet Etiquette - Please Read
body
```

```
      body
      body
```

The first line consisted of an "A" followed by an article ID (analogous to a message identifier and used for similar purposes). The second line was the list of newsgroups. The third line was the path. The fourth was the date, in the format above (all fields fixed width), resembling an Internet date but not quite the same. The fifth was the subject.

This format is documented for archeological purposes only. Articles MUST NOT be generated in this format.

Appendix A.2 - Early B-News Article Format

The obsolete pseudo-Internet article format, used briefly during the transition between the A News format and the modern format, followed the general outline of a MAIL message but with some non-standard headers. For example:

```
From: cbosgd!mhuxj!mhuxt!eagle!jerry (Jerry Schwarz)
Newsgroups: news.misc
Title: Usenet Etiquette -- Please Read
Article-I.D.: eagle.642
Posted: Fri Nov 19 16:14:55 1982
Received: Fri Nov 19 16:59:30 1982
Expires: Mon Jan 1 00:00:00 1990

body
body
body
```

The From-header contained the information now found in the Path-header, plus possibly the full name now typically found in the From-header. The Title-header contained what is now the Subject-content. The Posted-header contained what is now the Date-content. The Article-I.D.-header contained an article ID, analogous to a message identifier and used for similar purposes. The Newsgroups- and Expires-headers were approximately as now. The Received-header contained the date when the latest relaying agent to process the article first saw it. All dates were in the above format, with all fields fixed width, resembling an Internet date but not quite the same.

This format is documented for archeological purposes only. Articles MUST NOT be generated in this format.

Appendix A.3 - Obsolete Headers

Early versions of news software following the modern format sometimes generated headers like the following:

```
Relay-Version: version B 2.10 2/13/83; site cbosgd.UUCP
```

Posting-Version: version B 2.10 2/13/83; site eagle.UUCP
Date-Received: Friday, 19-Nov-82 16:59:30 EST

Relay-Version contained version information about the relaying agent that last processed the article. Posting-Version contained version information about the posting agent that posted the article. Date-Received contained the date when the last relaying agent to process the article first saw it (in a slightly nonstandard format).

In addition, this present standard obsoletes certain headers defined in [Son-of-1036] (see 6.22):

```
Also-Control: cancel <9urrt98y53@site.example>
See-Also: <i4g587y@site1.example> <kgb2231+ee@site2.example>
Article-Names: comp.foo:charter
Article-Updates: <i4g587y@site1.example>
```

Also-Control indicated a control message that was also intended to be filed as a normal article. See-Also listed related articles, but without the specific relationship with followups that pertains to the References-header. Article-Names indicated some special significance of that article in relation to the indicated newsgroup. Article-Updates indicated that an earlier article was updated, without at the same time being superseded.

These headers are documented for archeological purposes only. Articles containing these headers MUST NOT be generated.

Appendix A.4 - Obsolete Control Messages

This present standard obsoletes certain control messages defined in [RFC 1036] (see 7.5), all of which had the effect of requesting a description of a relaying or serving agent's software, or its peering arrangements with neighbouring sites, to be emailed to the article's reply address. Whilst of some utility when Usenet was much smaller than it is now, they had become no more than a tool for the malicious sending of mailbombs. Moreover, many organizations now consider information about their internal connectivity to be confidential.

```
version
sendsys
whogets
senduuname
```

"Version" requested details of the transport software in use at a site. "Sendsys" requested the full list of newsgroups taken, and the peering arrangements. "Who gets" was similar, but restricted to a named newsgroup. "Senduuname" resembled "sendsys" but restricted to the list of peers connected by UUCP.

Historically, a checkgroups body consisting of one or two lines, the first of the form "-n newsgroup", caused check-groups to apply to only that single newsgroup.

Historically, an article posted to a newsgroup whose name had exactly three components of which the third was "ctl" signified that article was to be taken as a control message. The Subject-header specified

the actions, in the same way the Control-header does now.

These forms are documented for archeological purposes only; they MUST NO LONGER be used.

Appendix B - Collected Syntax

Appendix B.1 - Characters, Atoms and Folding

In the following syntactic rules, numbers in the left hand margin indicate rules taken from other documents, specifically:

1 from [RFC 2231] having regard to errata contained in [RFC Errata];
2 from [RFC 2822] with the exception of those elements described therein as "obsolete";
3 from [RFC 2373];
4 from [RFC 2234];
5 from [RFC 2045].

Where the number is followed by an asterisk ('*'), it indicates that the rule in question has been modified for the purposes of this standard.

```
4  ALPHA               = %x41-5A /        ; A-Z
                         %x61-7A          ; a-z
2  CFWS                = *([FWS] comment) ( ([FWS] comment) / FWS )
4  CR                  = %x0D             ; carriage return
4  CRLF                = CR LF
4  DIGIT               = %x30-39          ; 0-9
4  DQUOTE              = %d34             ; quote mark
2  FWS                 = ([*WSP CRLF] 1*WSP); folding whitespace
4  HEXDIG              = DIGIT / "A" / "B" / "C" / "D" / "E" / "F"
4  HTAB                = %x09             ; horizontal tab
4  LF                  = %x0A             ; line feed
2  NO-WS-CTL           = %d1-8 /          ; US-ASCII control characters
                         %d11 /           ; which do not include the
                         %d12 /           ; carriage return, line feed,
                         %d14-31 /        ; and whitespace characters
                         %d127
4  SP                  = %x20             ; space
4  WSP                 = SP / HTAB        ; whitespace characters
2  atext               = ALPHA / DIGIT /
                         "!" / "#" /      ; Any character except
                         "$" / "%" /      ; controls, SP, and specials.
                         "&" / "'" /      ; Used for atoms
                         "*" / "+" /
                         "-" / "/" /
                         "=" / "?" /
                         "^" / "_" /
```

```
                        "`" / "{" /
                        "|" / "}" /
                        "~"
2  atom                 = [CFWS] 1*atext [CFWS]
2* ccontent             = ctext / quoted-pair / comment / encoded-word
```

```
   charset              = <registered character set name>
                                        ;  [RFC 2978]
2  comment              = "(" *([FWS] ccontent) [FWS] ")"
2  ctext                = NO-WS-CTL /      ; all of <text> except
                          %d33-39 /        ; SP, HTAB, "(", ")"
                          %d42-91 /        ; and "\"
                          %d93-126
1  encoded-word         = "=?" charset ["*" language] "?" encoding
                             "?" encoded-text "?="
2  dcontent             = dtext / quoted-pair
2  dot-atom             = [CFWS] dot-atom-text [CFWS]
2  dot-atom-text        = 1*atext *( "." 1*atext )
2  dtext                = NO-WS-CTL /      ; Non white space controls
                          %d33-90 /        ; The rest of the US-ASCII
                          %d94-126         ; characters not including
                                           ; "[", "]", or "
   extended-phrase      = ( [CFWS] encoded-word [CFWS] / word )
                             *( [CFWS] encoded-word [CFWS] / word  /
                                [CFWS] "." [CFWS] )
   language             = <registered language tag>
                                        ;  [RFC 3066]
2* phrase               = 1*( [CFWS] encoded-word [CFWS] / word ) /
                          extended-phrase
2  qcontent             = qtext / quoted-pair
2  qtext                = NO-WS-CTL /      ; all of <text> except
                          %d33 /           ; SP, HTAB, "\" and DQUOTE
                          %d35-91 /
                          %d93-126
2  quoted-pair          = "\" text
2  quoted-string        = [CFWS] DQUOTE
                             *( [FWS] qcontent ) [FWS]
                             DQUOTE [CFWS]
2  specials             = "(" / ")" /      ; Special characters used in
                          "<" / ">" /      ;  other parts of the syntax
                          "[" / "]" /
                          ":" / ";" /
                          "@" / "\" /
                          "," / "." /
                          DQUOTE
2  text                 = %d1-9 /          ; all US-ASCII characters except
                          %d11-12 /        ; NUL, CR and LF
                          %d14-127
5  tspecials            = "(" / ")" / "<" / ">" / "@" /
                          "," / ";" / ":" / "\" / DQUOTE /
                          "/" / "[" / "]" / "?" / "="
2* unstructured         = 1*( [FWS] ( utext / encoded-word ) ) [FWS]
2  utext                = NO-WS-CTL /      ; Non white space controls
                          %d33-126         ; The rest of US-ASCII
2  word                 = atom / quoted-string
```

Appendix B.2 - Basic Forms

```
2  addr-spec            = local-part "@" domain
2  address              = mailbox / group
2  address-list         = address *( "," address )
2  angle-addr           = [CFWS] "<" addr-spec ">" [CFWS]
   article              = 1*( header CRLF ) separator body
1  attribute            = 1*attribute-char
1  attribute-char       = <any (US-ASCII) CHAR except SPACE, CTLs,
                           "*", "'", "%", or tspecials>
   body                 = *( *998text CRLF )
2  display-name         = phrase
2  date                 = day month year
2  date-time            = [ day-of-week "," ] date FWS time [CFWS]
2  day                  = [FWS] 1*2DIGIT
2  day-name             = "Mon" / "Tue" / "Wed" / "Thu" /
                          "Fri" / "Sat" / "Sun"
2  day-of-week          = [FWS] day-name
2  domain               = dot-atom / domain-literal
2  domain-literal       = [CFWS] "[" *([FWS] dcontent) [FWS] "]" [CFWS]
1  extended-initial-name
                        = attribute [initial-section] "*"
1  extended-initial-value
                        = [charset] "'" [language] "'"
                          extended-other-values
1  extended-other-names = attribute other-sections "*"
1* extended-other-values= *( "%" 2HEXDIG / attribute-char )
1* extended-parameter   = ( [CFWS] extended-initial-name [CFWS]
                           "=" extended-initial-value ) /
                          ( [CFWS] extended-other-names [CFWS]
                           "=" extended-other-values )
   extension-parameter  = <a parameter not defined by this standard>
2  group                = display-name ":" [ mailbox-list / CFWS ] ";"
                           [CFWS]
   header-name          = 1*name-character *( "-" 1*name-character )
2  hour                 = 2DIGIT
1  initial-section      = "*0"
2  local-part           = dot-atom / quoted-string
2  mailbox              = name-addr / addr-spec
2  mailbox-list         = mailbox *( "," mailbox )
2  minute               = 2DIGIT
2  month                = FWS month-name FWS
2  month-name           = "Jan" / "Feb" / "Mar" / "Apr" /
                          "May" / "Jun" / "Jul" / "Aug" /
                          "Sep" / "Oct" / "Nov" / "Dec"
2  name-addr            = [display-name] angle-addr
   name-character       = ALPHA / DIGIT
   other-header         = header-name ":" 1*SP other-content
   other-content        = <the content of a header defined by some
```

```
                          other standard>
   other-section        = "*" ("1" / "2" / "3" / "4" / "5" /
                           "6" / "7" / "8" / "9") *DIGIT
1  parameter            = regular-parameter / extended-parameter
```

```
1* regular-parameter    = [CFWS] regular-parameter-name [CFWS]
                             "=" value
1  regular-parameter-name
                        = attribute [section]
2  second               = 2DIGIT
1  section              = initial-section / other-section
   separator            = CRLF
2  time                 = time-of-day FWS zone
2  time-of-day          = hour ":" minute [ ":" second ]
5  token                = 1*<any (US-ASCII) CHAR except SP, CTLs,
                             or tspecials>
5  value                = [CFWS] token [CFWS] / quoted-string
   x-attribute          = "x-" attribute
2  year                 = 4*DIGIT
2* zone                 = (( "+" / "-" ) 4DIGIT) / "UT" / "GMT"
```

Appendix B.3 - Headers

Appendix B.3.1 - Header outlines

```
   header               = other-header /
                          Date-header /
                          From-header /
                          Message-ID-header /
                          Subject-header /
                          Newsgroups-header /
                          Path-header /
                          Injection-Date-header /
                          Reply-To-header /
                          Sender-header /
                          Organization-header /
                          Keywords-header /
                          Summary-header /
                          Distribution-header /
                          Followup-To-header /
                          Mail-Copies-To-header /
                          Posted-And-Mailed-header /
                          References-header /
                          Expires-header /
                          Archive-header /
                          Control-header /
                          Approved-header /
                          Supersedes-header /
                          Xref-header /
                          Lines-header /
                          User-Agent-header /
                          Injection-Info-header /
                          Complaints-To-header
```

```
   Approved-content     = From-content
   Approved-header      = "Approved" ":" SP Approved-content
                             *( ";" extension-parameter )
   Archive-content      = [CFWS] ("no" / "yes" ) [CFWS]
```

```
Archive-header       = "Archive" ":" SP Archive-content
                          *( ";" ( Archive-parameter /
                                   extension-parameter ) )
Archive-parameter    = <a parameter with attribute "filename"
                        and any value>
Complaints-To-content= address-list
Complaints-To-header = "Complaints-To" ":" SP Complaints-To-content
Control-content      = [CFWS] control-message [CFWS]
Control-header       = "Control" ":" SP Control-content
                          *( ";" extension-parameter )
Date-content         = date-time
Date-header          = "Date" ":" SP Date-content
Distribution-content = distribution *( dist-delim distribution )
Distribution-header  = "Distribution" ":" SP Distribution-content
                          *( ";" extension-parameter )
Expires-content      = date-time
Expires-header       = "Expires" ":" SP Expires-content
                          *( ";" extension-parameter )
Followup-To-content  = Newsgroups-content /
                       [FWS] %x70.6F.73.74.65.72 [FWS]
                       ; which is a case-sensitive "poster"
Followup-To-header   = "Followup-To" ":" SP Followup-To-content
                          *( ";" extension-parameter )
From-content         = mailbox-list
From-header          = "From" ":" SP From-content
Injection-Date-content
                     = date-time
Injection-Date-header
                     = "Injection-Date" ":" SP Injection-Date-content
                          *( ";" extension-parameter )
Injection-Info-content
                     = [CFWS] path-identity [CFWS]
Injection-Info-header= "Injection-Info" ":" SP Injection-Info-content
                          *( ";" ( Injection-Info-parameter /
                                   extension-parameter ) )
Injection-Info-parameter
                     = posting-host-parameter /
                       posting-account-parameter /
                       posting-sender-parameter /
                       posting-logging-parameter
Keywords-content     = phrase *( "," phrase )
Keywords-header      = "Keywords" ":" SP Keywords-content
Lines-content        = [CFWS] 1*DIGIT [CFWS]
Lines-header         = "Lines" ":" SP Lines-content
                          *( ";" extension-parameter )
Mail-Copies-To-content
                     = copy-addr / [CFWS] ( "nobody" /
                                   "poster" ) [CFWS]
Mail-Copies-To-header= "Mail-Copies-To" ":" SP Mail-Copies-To-content
```

```
   Message-ID-content   = [FWS] msg-id [FWS]
   Message-ID-header    = "Message-ID" ":" SP Message-ID-content
   Newsgroups-content   = [FWS] newsgroup-name
                            *( [FWS] ng-delim [FWS] newsgroup-name )
                            [FWS]
```

```
   Newsgroups-header    = "Newsgroups"  ":" SP Newsgroups-content
                               *( ";" extension-parameter )
   Organization-content = unstructured
   Organization-header  = "Organization" ":" SP Organization-content
   Path-content         = [FWS] *( path-identity [FWS]
                                   path-delimiter [FWS]
                              ) tail-entry [FWS]
   Path-header          = "Path" ":" SP Path-content
                           *( ";" extension-parameter )
   Posted-And-Mailed-content
                        = [CFWS] ( "yes" / "no" ) [CFWS]
   Posted-And-Mailed-header
                        = "Posted-And-Mailed" ":" SP
                             Posted-And-Mailed-content
                             *( ";" extension-parameter )
   References-content   = [CFWS] msg-id *( CFWS msg-id ) [CFWS]
   References-header    = "References" ":" SP References-content
   Reply-To-content     = address-list
   Reply-To-header      = "Reply-To" ":" SP Reply-To-content
   Sender-content       = mailbox
   Sender-header        = "Sender" ":" SP Sender-content
   Subject-content      = unstructured
   Subject-header       = "Subject" ":" SP Subject-content
   Summary-content      = unstructured
   Summary-header       = "Summary" ":" SP Summary-content
   Supersedes-content   = [CFWS] msg-id [CFWS]
   Supersedes-header    = "Supersedes" ":" SP Supersedes-content
   User-Agent-content   = product *( CFWS product )
   User-Agent-header    = "User-Agent" ":" SP User-Agent-content
                           *( ";" extension-parameter )
   Xref-content         = [CFWS] server-name 1*( CFWS location ) [CFWS]
   Xref-header          = "Xref" ":" SP Xref-content
                           *( ";" extension-parameter )
```

Appendix B.3.2 - Control-message outlines

```
   control-message      = <empty> /
                          Newgroup-message /
                          Rmgroup-message /
                          Mvgroup-message /
                          Checkgroup-message /
                          Cancel-message /
                          Ihave-message /
                          Sendme-message

   Cancel-arguments     = CFWS msg-id [CFWS]
   Cancel-message       = "cancel" Cancel-arguments
   Checkgroup-arguments = [ chkscope ] [ chksernr ]
   Checkgroup-message   = "checkgroups" Checkgroup-arguments
```

```
   Ihave-arguments      = relayer-name
   Ihave-message        = "ihave" Ihave-arguments
   Mvgroup-arguments    = CFWS newsgroup-name CFWS newsgroup-name
                            [ CFWS newgroup-flag ]
   Mvgroup-message      = "mvgroup" Mvgroup-arguments
```

```
   Newgroup-arguments   = CFWS newsgroup-name [ CFWS newgroup-flag ]
   Newgroup-message     = "newgroup" Newgroup-arguments
   Rmgroup-arguments    = CFWS newsgroup-name
   Rmgroup-message      = "rmgroup" Rmgroup-arguments
   Sendme-arguments     = Ihave-arguments
   Sendme-message       = "sendme" Sendme-arguments
```

Appendix B.3.3 - Other header rules

```
   article-locator      = 1*( %x21-27 / %x29-3A / %x3C-7E )
                               ; US-ASCII printable characters
                               ; except'(' and ';'
   article-size         = 1*DIGIT
   batch                = 1*( batch-header article )
   batch-header         = "#!" SP rnews SP article-size CRLF
   checkgroups-body     = *( valid-group CRLF )
   chkscope             = 1*( CFWS ["!"] newsgroup-name )
   chksernr             = CFWS "#" 1*DIGIT
   component            = 1*component-grapheme
   component-grapheme   = DIGIT / ALPHA / "+" / "-" / "_"
   copy-addr            = address-list
   dist-delim           = ","
   distribution         = [FWS] distribution-name [FWS]
   distribution-name    = ALPHA 1*distribution-rest
   distribution-rest    = ALPHA / "+" / "-" / "_"
   groupinfo-body       = [ newsgroups-tag CRLF ]
                             newsgroups-line CRLF
3  hex4                 = 1*4HEXDIG
3  hexpart              = hexseq / hexseq "::" [ hexseq ] /
                          "::" [ hexseq ]
3  hexseq               = hex4 *( ":" hex4 )
   host-value           = dot-atom /
                          [ dot-atom ":" ]
                            ( IPv4address / IPv6address )
                            ; see  [RFC 2373]
2  id-left              = dot-atom-text / no-fold-quote
2  id-right             = dot-atom-text / no-fold-literal
   ihave-body           = *( msg-id CRLF )
3  IPv4address          = 1*3DIGIT "." 1*3DIGIT "."
                             1*3DIGIT "." 1*3DIGIT
3  IPv6address          = hexpart [ ":" IPv4address ]
   location             = newsgroup-name ":" article-locator
   mdtext               = NO-WS-CTL /      ; Non white space controls
                          %d33-61 /        ; The rest of the US-ASCII
                          %d63-90 /        ; characters not including
                          %d94-126         ; ">", "[", "]", or "\"
   moderation-flag      = %x28.4D.6F.64.65.72.61.74.65.64.29
                             ; case sensitive "(Moderated)"
```

```
   mqspecial            = "(" / ")" /      ; same as specials except
                          "<" /            ; "\" and DQUOTE quoted
                          "[" / "]" /      ; and ">" omitted
                          ":" / ";" /
                          "@" / "\\" /
                          "," / "." /
                          "\" DQUOTE
   mqtext               = NO-WS-CTL /      ; all of <text> except
                          %d33 /           ; SP, HTAB, "\", ">"
                          %d35-61 /        ; and DQUOTE
                          %d63-91 /
                          %d93-126
2* msg-id               = "<" id-left "@" id-right ">"
   newgroup-flag        = "moderated"
   newsgroup-description
                        = utext *( *WSP utext )
   newsgroup-name       = component *( "." component )
   newsgroups-line      = newsgroup-name
                             [ 1*HTAB newsgroup-description ]
                             [ 1*WSP moderation-flag ]
   newsgroups-tag       = %x46.6F.72 SP %x79.6F.75.72 SP
                             %x6E.65.77.73.67.72.6F.75.70.73 SP
                             %x66.69.6C.65.3A
                             ; case sensitive
                             ; "For your newsgroups file:"
   ng-delim             = ","
2* no-fold-literal      = "[" *( mdtext / "\[" / "\]" / "\\" ) "]"
2* no-fold-quote        = DQUOTE
                             *( mqtext / "\\" / "\" DQUOTE )
                             mqspecial
                             *( mqtext / "\\" / "\" DQUOTE )
                             DQUOTE
   path-delimiter       = "/" / "?" / "%" / "," / "!"
   path-identity        = ( ALPHA / DIGIT )
                             *( ALPHA / DIGIT / "-" / "." / ":" / "_" )
   posting-account-parameter
                        = <a parameter with attribute "posting-account"
                           and any value>
   posting-host-parameter
                        = <a parameter with attribute "posting-host"
                           and value some host-value>
   posting-logging-parameter
                        = <a parameter with attribute "logging-data"
                           and any value>
   posting-sender-parameter
                        = <a parameter with attribute "sender"
                           and value some sender-value>
   product              = [CFWS] token [CFWS] [ "/" product-version ]
   product-version      = [CFWS] token [CFWS]
```

```
   relayer-name         = path-identity
   rnews                = %x72.6E.65.77.73 ; case sensitive "rnews"
   sender-value         = mailbox / "verified"
   sendme-body          = ihave-body
   server-name          = path-identity
```

```
tail-entry          = path-identity
valid-group         = newsgroups-line
```

Appendix C - Notices

Intellectual Property

The IETF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on the IETF's procedures with respect to rights in standards-track and standards-related documentation can be found in BCP-11. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

Full Copyright Statement

Copyright (C) The Internet Society (2003). All Rights Reserved

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING

TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.