IEEE P802.11  
Wireless LANs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 802.11 Style Guide | | | | |
| Date: 2020-11 | | | | |
| Author(s): | | | | |
| Name | Company | Address | Phone | email |
| Robert Stacey | Intel |  |  | robert.stacey@intel.com |
| Adrian Stephens | Intel Corporation |  |  | Adrian.p.stephens@intel.com |
| Peter Ecclesine | Cisco Systems | 170 W. Tasman Dr., San Jose, CA 95134-1706 | +1-408-527-0815 | petere@cisco.com |
| William Marshall | AT & T |  |  | wtm@research.att.com |
| Emily Qi | Intel |  |  | Emily.h.qi@intel.com |
| Edward Au | Huawei Technologies |  |  | edward.ks.au@gmail.com |

Abstract

This document contains a description of certain elements of style to be used in 802.11 Standards and Amendments, starting with REVmb (IEEE STD 802.11-2011) and its amendments.

R1: updated reflecting additional changes in REVmb. Numbering consistent with REVmb\_D8.0.

R2: added comments about word usage from experience gained during 802.11aa 11MEC

R3: updated to include items from 11-11-0844r0. Added section on “ensures”.

R4: updated following Std 802.11-2012 publication.

R5: typos fixed. Added “changes in IEEE-SA style guide” section

R6: reviewed prior to 802.11ac MDR.

R7: updated based on REVmc pre-ballot comment resolutions.

R8: updates from TGaf MDR

R9: updates from REVmc D1 comment resolution

R10: changes from REVmc D3 MDR (11-14/781r12) and reviewed

R11: changes from TGah and TGai MDR

R12: new amendment style (top level clause for major MAC enhancements). Update following 802.11-2016 publication.

R13: added new recommendations on MIB objects. Changes from REVmd MDR.

R14: update the list of exceptions in section 2.11

R15: added section “ 2.4.3 Element Inclusion Conventions” and the procedure of deprecating a MIB variable in 3.4.2.

R16: Added rule to capitalize prepositions and conjuctions in proper names

R17: updates from REVmd 5.0.

# Introduction

## Purpose

The purpose of this document is to describe certain elements of style to be used in IEEE 802.11 Standards and Amendments starting with IEEE Std 802.11-2016, and including its amendments.

There are two benefits from using this style:

* Uniformity/consistency internally and between different documents. Don’t forget that amendments eventually are edited into the baseline in the next revision standard. If the amendment uses styles that are inconsistent with the baseline, we either have to tolerate an inconsistent revision, or the editor/comment resolution committee have to resolve those inconsistencies.
* To reduce the amount of work in committee – i.e., these decisions have already been taken, and there is no need for the editor/comment resolution committee of a new document to have to re-invent the wheel.

In addition, it is expected that this document may be cited in comment resolutions where a standard/amendment does not comply with this style.

## WG802.11 style vs IEEE-SA style

What right do we have to define an 802.11 Style? Surely the IEEE-SA Style guide is all we need?

There are many elements of convention that are not addressed in the IEEE-SA Style guide where we benefit from consistency in 802.11.

There are also elements of style in 802.11 that fail to comply with the IEEE-SA Style guide. The fact that this material has been through IEEE-SA publication editing and approved multiple times shows that slavish consistency to the IEEE-SA Style Guide is not a requirement of the IEEE-SA standards development process.

However, gratuitous deviation from the IEEE-SA Style Guide is not a good idea.

These principles are embodied in the following requirement:

* An 802.11 Revision or Amendment shall comply with this document. An 802.11 Revision or Amendment shall comply with the IEEE-SA Style Guide.
  + Where there is a conflict between these two documents, this document takes precedence.
  + Where there is a clear and consistent conflict between the IEEE-SA Style Guide and the 802.11 baseline standard, this should be discussed (e.g., in the Editors’ meeting or TGm<letter>) and this document updated.

This document should also be updated by the Editors’ or TGm<letter> when new editorial conventions are decided.

## WG 802.11 MDR

802.11 will conduct a mandatory draft review (MDR) between the document editor and the WG802.11 technical editor(s).

One purpose of this MDR is to ensure compliance with the requirements of this document.

The MDR is described in 11-11/0615.

## ANA Process

Documents that need numbers from namespaces administered by the ANA should insert <ANA> into their document in place of a number rather than choosing a number.

The editor will then ask the ANA for numbers to be assigned (see 802.11 Operations Manual for full procedure), and when numbers have been assigned, the editor should replace the <ANA> flags with the allocated values.

All <ANA> flags must have been replaced before entry to sponsor ballot.

## Why can’t I leave all this to the publication editor?

The argument goes that seeing that the IEEE-SA pays for a professional editor to clean-up the document prior to publication, we (i.e., contributors to drafts and their TG editors) don’t need to worry about elements of “style” in the draft, particularly as we are engineers and not professional editors, and can’t be expected to understand or care about obscure rules of grammar.

But if you do this:

1. The WG MDR will be a painful experience (for all concerned)
2. You may continually get lots of editorial comments, which take more effort to deal with than getting things right earlier in the process
3. Publication editing will be a painful and slow experience (for the TG editor, who bears the burden of review)
4. Publication editors vary in the degree of diligence they apply to the task, particularly related to uniformity of style with the baseline.
5. Adopting a uniform style creates “least surprise” amongst your readers.
6. Poor and broken English creates the wrong impression for your voters related to completeness or technical accuracy.
7. Rules about, punctuation and. syntax matter – they can modify the meaning of the text;

# General 802.11 Style

## Frames

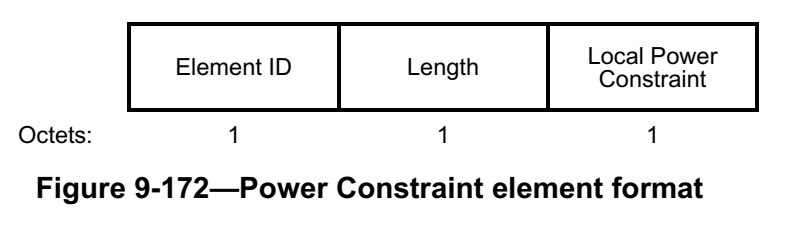
### Frame Format Figures

See 11-09-0714-02-000m-big-ed-issues.doc for more detail and background.

Generally frame format figures should be either an “octet aligned” or a “bit aligned” structure. If a mixture of the two is required, it is recommended to break the figure into two or more parts. Arrows should not be used, except where labelling parts of the structure (e.g., the MAC Header in the top-level frame format).

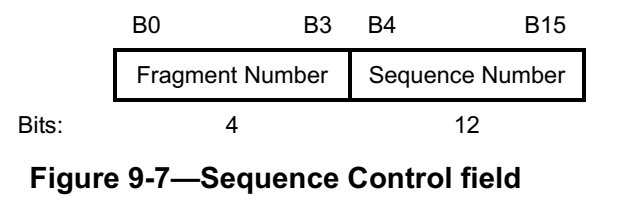
Figures should be drawn using tables with each part of the figure in its own cell, along with custom ruling. The bit labels above a cell can be created by inserting a right tab marker at the right margin (e.g., using the Framemaker ruler bar), then inserting “B<number><tab>B<number>” and set the justification to left. This is preferred to using spaces. The ruling can be created in frame by selecting all the cells and applying custom ruling “none”, then select the cells containing the fields and apply custom ruling “thin”.

An example of the octet-aligned figure follows:



Another used format is those figures where the “octets” row goes above the structure boxes to allow for arrows and labelling below the structure boxes. Such uses should be minimized in new material.

An example of the bit-aligned figure follows:



The title of the figure should use the form “Figure <number>--Name [frame, element, field, subelement or subfield] format”.

#### Optional Fields

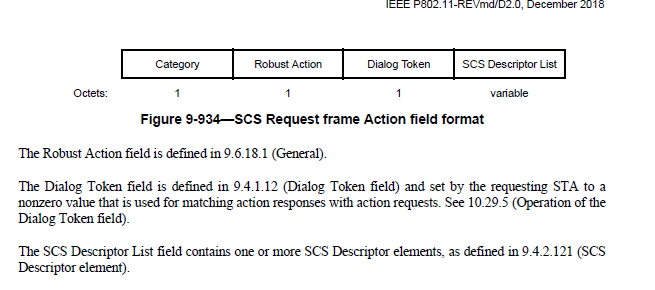
Where an octet field is optional (i.e., may or may not be present), it shall be shown as “0 or <n>”, or “variable” in the “Octets:” entry.

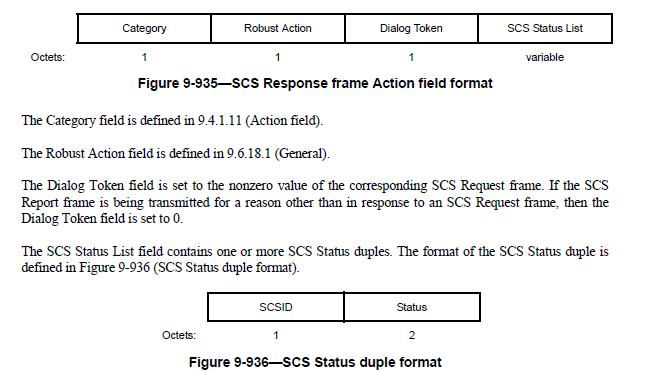
#### Repeating Fields

Repeating fields should be avoided in the frame/element/field format figure. If a field needs to be repeated, create a container field with “List” in the name, e.g., Name List field.

Describe the list field as follows: The Name List field contains {one, zero} or more Name fields.

Two examples of the repeating field follow:

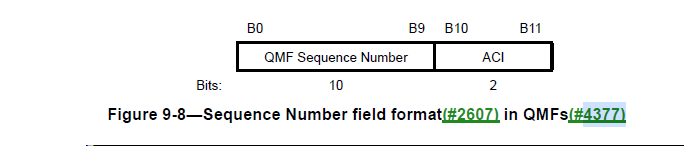


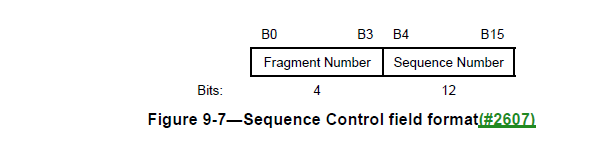


#### Bit Position

In a bit-aligned figure, the bit position should always start with B0.

For example, the bit position in the Sequence Number field format should start with B0, although the Sequence Number field shown in the Sequence Control field format starts with B4.





### Naming Frames

A “formal” reference to a frame (i.e., that used in any normative statement) should follow the pattern “<name of frame> frame”, with the name Initial Caps and “frame” in lower caps.

So, “shall transmit an Association Request frame” is correct. But the following are incorrect:

* shall transmit an Association Request
* … Association request frame
* … Association Request Management frame
* shall transmit a Management frame of subtype Association Request

Unless “request”, “response”, “Management”, “Data”, “Action” are part of the name of the frame, they should not be included.

Note, the following correct uses:

* a Management frame
* a Data frame
* a Control frame
* an Extension frame
* an Action frame

Note that the full name of the frame shall always be used. So “An ADDBA frame” is incorrect, because the actual names of the frames are ADDBA Request and ADDBA Response. If you need to indicate both types of frame, explicitly cite both frames.

## Case of true/false

All “true” and “false” should be lower case, except where they are part of pseudo-code or code.

Note that MIB variables take “true” and “false” even in pseudo-code.

Note that the REVmc D3.0 MDR determines that the 802.1X state machine description can use “true” and “false” as 802.1X is itself inconsistent about whether its state machine variables are upper or lower case.

## “Is set to”

The verb “set” should only be used when describing how a field obtains a value, e.g. “The Measurement Duration field is set to the preferred or mandatory duration of the requested measurement, expressed in units of TUs.”

Where the value of the field is read or referenced, (e.g., in the context of a condition), “is set to” shall not be used.

Note that when a field value is tested in order to construct another field value, “equal to” is used for the test, and “set to” for the constructed field. “If the <xyz> field is equal to 0, the <abc> field shall be set to 1.”

## Information Elements/Subelements

Note that statements in this section also apply (with appropriate adjustment to wording and reference) to subelements.

### Naming

Elements should be called the “<Purpose> element”, where <Purpose> does not include the word “information” (e.g., the “QoS Capability element”). References to the structure from the text should include the word “element” at the end.

The phrase “information element” shall not appear.

The same convention exists for “subelements”.

### Definition Conventions

The structure of an element is defined to include an Element ID and Length field, common to all elements. Different groups found increasingly bizarre and tortuous ways of repeating this information per-element, such as:

* The value of the Element ID field is set to the value of the value column from the row from Table 8-26 whose name matches “xyz”.

There is no need to specify any value for the Element ID field.

There is no need to specify any value for the Length field, unless it is necessary to convey semantics over and above that stated for the generic format. An example might be if the value of the Length field is always a prime number less the phase of the moon.

The following statement: “The Element ID and Length fields are defined in 8.4.2.1 (General).” just after the element figure might be helpful to readers, and might forestall ill-advised letter ballot comments “the Length field needs a definition”.

### Element Inclusion Conventions

When defining a new element, as a general rule, do not list the frames that carry the element as part of element definition, and only list the element in the body of each of those frames that can include the element. Listing the frames that can carry the element is duplicate information (it can be inferred from frame definitions), and subject to inconsistencies over iterations of the specifications. Notable exception is when element definition depends on the frame it is carried in.

Examples from REVmd draft 2.4

Good style; no mention of frames that can carry the element.

**9.4.2.161 Transmit Power Envelope element**

The Transmit Power Envelope element conveys the local maximum transmit power for various transmission bandwidths. The format of the Transmit Power Envelope element is shown in Figure 9-614 (Transmit Power Envelope element format).

Bad style; redundant and potentially inconsistent information. Listing Wide Bandwidth Channel Switch element in each of the frames mentioned in the paragraph is sufficient.

**9.4.2.160 Wide Bandwidth Channel Switch element**

The Wide Bandwidth Channel Switch element is included in Channel Switch Announcement frames, as described in 9.6.2.6 (Channel Switch Announcement frame format), Extended Channel Switch Announcement frames, as described in 9.6.7.7 (Extended Channel Switch Announcement frame format), and TDLS Channel Switch Request frames, as described in 9.6.12.7 (TDLS Channel Switch Request Action field format).

## Removal of functions and features

Functions and features described in the published 802.11 standard shall not be removed unless they have been marked “obsolete and subject to removal in a subsequent revision of this standard.” in a previous revision.

Functions and features might also be “deprecated”, such as “The use of the dual CTS mechanism

is deprecated.”

These terms have slightly different functions. “Obsolete” is used as a route to remove unused and useless mechanisms. “Deprecated” is used as a method to indicate that implementations should stop using a feature, but acknowledges that the feature has been used and might still be used by existing implementations, and does not imply a feature’s future removal from the standard.

## Capitalization

Capital letters are often over-used. In particular, some groups seem to think that every concept they create (especially the Really Important Ones) deserves capitals. This Is Not So. Initial caps are used as described below.

Capital letters should be reserved for:

* Abbreviations
* Proper names of entities outside 802.11. Generally follow whatever appears to be the prevailing custom.
* Initial first letter of headings
* Initial letters of proper names, which includes:
  + Frame names – e.g. the “Beacon frame”, but “transmits a beacon” where it is used to represent the concept of a beacon (no caps).
  + Element & subelement names – e.g. “the Capabilities element”, but “the capabilities of the STA”
  + Field & subfield names – e.g. “the More Data subfield of the Frame Control field”, but “shall set it to 1 if it has more data to send”.
  + Enumerated values of a field or subfield
  + Certain measurement requests and reports (see 8.4.2.20 and 8.4.2.21)
* Grandfathered terms where the cost of changing the capitalization of a widely used term was considered excessive. The grandfathered terms include:
  + TIM Broadcast

For proper names, all words in the name (excluding acronyms) should be capitalized, including prepositions and conjunctions. This is to avoid ambiguity where the preposition or conjuction might be misinterpreted as part of the sentence. For example, “HT And VHT Trigger Frame RX Support subfield.” This rule may not apply to legacy text; these remain unchanged since the effort involved in changing all instances is substantial and no harm has been demonstrated.

Note “Block Ack” was removed as a grandfathered term in REVmc. “Block Ack” still exists in this form, but only where it occurs in the name of a field or as an enumeration value.

## Terminology: frame vs packet vs PPDU vs MPDU

“Frame” is interpreted based on context. In references to MAC structures, “frame” is synonymous with “MPDU”. In references to PHY structures, “frame” is synonymous with PPDU.

Generally, “frame” is preferred to MPDU, particularly when it is a named frame type. For example: “Ack frame”, not “Ack MPDU”.

Generally, PPDU is preferred to frame in the PHY.

The use of “packet” should be minimized, except where this terminology is defined by external specifications, e.g. IETF.

## Use of verbs & problematic words

The normative verbs are:

* shall - equivalent to “is required to”, “has to”, “must”
* should – equivalent to “is recommended to”, “is advised to”
* may – equivalent to “is allowed to”, “is permitted to”

The non-normative verbs are:

* can – equivalent to “is able to” or “is allowed to, as defined elsewhere in this standard” (see below)
* might – equivalent to “chooses according to unspecified criteria”

Normative verbs shall not appear in informative text. The usual culprits are verbs like “may” or “should” that appear in NOTEs. This is also true for equivalent phrases shown above

Note that TGmc has muddied the waters somewhat. It decided that “should” can appear in informative text, such as NOTES and Annexes. The rationale is that “should” is a recommendation, but it is entirely up to the implementer whether to pay attention to the recommendation or not. As it therefore doesn’t necessarily affect an implementation, it should not affect interoperable behaviour and is therefore informative.

Notwithstanding the above, the use of “should” in informative material should be minimized.

Verbs deprecated by the IEEE (e.g., “must”, “will”) should not be used, except:

* “will” can be used when stating future fact, and using any other tense dilutes the intended meaning. This case should be rare.
* “must” is used in some boilerplate reproduced from the IEEE-SA style guide. Do not edit the boilerplate text ;0).
* “must” might also be used when regulation is quoted verbatim.

The use of “can” should be considered carefully. It should be interpreted as meaning “is allowed to” or “is able to”, when interpreted in the context of normative statements made elsewhere in the standard, or referenced from the standard, or that are generally well known or self-obvious. When those conditions do not exist, use the word “might” as this expresses the possibility of something happening without use understanding how it can happen.

“May not” is ambiguous. This phrase should be replaced by “shall not”, or possibly (“may <x> if <y>; shall not <x> if not <y>”.

“Only” is often misused. “Only” is a constraint, which should apply to a condition, not to a verb (unless that verb is part of expressing a condition).

So: “A STA shall only transmit an Ack when it receives a packet” is wrong, and “A STA shall transmit an Ack only when it receives a packet” is right.

Also “shall … only” often doesn’t go far enough, and worse, is interpreted differently by different readers. “A STA shall transmit an Ack only when it receives a packet” can be interpreted as “A STA shall not transmit an Ack when does not receive a packet” – i.e. expressing only the constraint. A better unambiguous rule is: “A STA shall transmit an Ack when it receives a packet; it shall not transmit an ACK otherwise.” – which makes both the positive (do this) and negative (don’t do this) conditions explicit.

*Bottom line:* minimize use of “only” in normative text.

Replace “shall <a> only if <b>” with an alterntive, such as one of the following forms:

* “shall not <a> if not <b>”
* “may <a> if <b>; shall not <a> if not <b>”
* “shall <a> if <b>; shall not <a> if not <b>”

“Ensures” should be avoided, and although the IEEE-SA style guide has nothing to say on this topic, the IEEE-SA editors reword to avoid its use, as it may be construed as providing a guarantee. Can be reworded as “verify that <condition> is true, and take remedial action if not.”

REVmc D3.0 MDR determined that “shall … ensure” is OK, e.g. “A STA shall ensure that all transmissions and any responses fit within the TXOP duration.” The objection to “ensure” is that it might hide a normative requirement. Howevever in “shall … ensure”, the normative requirement is explicit.

Note that these rules do not apply in material that is a direct quote from another source. However note also that the IEEE-SA will require copyright letters from any copyright owners whose material is directly quoted.

### Which / that

Note that the use of which and that as described in the IEEE-SA style guide.

E.g., use of “which” should generally be preceded by a comma or preposition.

### Missing & use of articles

A common error in non-English native speakers is improper use of articles: “a, an, the”.

These are important, because they identify they identify which entity or under what conditions behaviour is being described. An improper article can create ambiguity about specification, so this should not be considered something to “leave to the publication editor”.

The errors are:

* Missing article
* Improper article (“a” vs “an”)
* Missing antecedent (“the” without a preceding identification).

The following example is typical:

* A STA that receives an abc frame shall transmit an xyz response frame.
* The xyz response frame shall have the def field set to 1.
* The STA shall wait for another abc frame for up to aWaitTime, and if no such frame arrives, explode in a puff of logic.

### Missing noun in noun phrase

Many of the things we define take part in a noun phrase. For example,

A Beacon frame

The HT Capabilities element

The HT Capabilities Info field [of the HT Capabilities element]

The HT-Greenfield subfield [of the The HT Capabilities Info field [of the HT Capabilities element]]

A PHY-TXSTART.confirm primitive.

The Length field.

Do not miss out the “noun” part of the noun phrase. i.e., “The Length field is set to 1” is correct, but “The Length is 1” is incorrect.

### Unnecessary noun in noun phrase

The following terms are already noun phrases and do not need any following nouns:

* A SIFS (and all the other IFS values). There is no need to follow with “spacing”/”period”/”Interval”.

### Unicast and Multicast

When used to describe MAC entities, the adjectives “unicast” and “directed” are deprecated in favor of “individually addressed” or “that is an individual address” and the adjective “multicast” is deprecated in favor of “group addressed” or “that is a group address.

* To modify a noun such as frame, MPDU, MSDU, PPDU, etc., or a conceptual object such MAC entity, use “individually addressed frame” or <example with MAC entity>.
* To describing a type of address, use “that is an individual address” or “that is a group address.” For example, “destination address that is an individual address” or “RA that is a group address.”

The terms “unicast” (or “directed”) and “multicast” may be used with non-MAC entities, for example:

* Unicast communication
* MIB object names
* directed multicast service
* Flexible multicast service
* Multicast parameters for FMS Request
* Multicast Diagnostic
* FMS multicast rate
* multicast integrity protocol
* Multicast Triggered Reporting
* multicast group
* multicast reception
* multicast traffic
* broadcast/multicast transmitter,

### Normative and Informative

The IEEE-SA Standards Board Operations Manual defines which parts of a standard are normative and which parts of a standard are informative. Normative text is information that is required to implement the standard and is therefore officially part of the standard. Informative text is provided for information only and is therefore not officially part of the standard.

Normative text (information required to implement the standard) includes the following:

* The main clauses of the documents including figures, tables, and equations
* Footnotes to tables

Interspersed normative and informative text is not allowed. As such,

* Neither clauses nor subclauses shall be labeled as informative.
* Figures or tables in clauses shall not be labeled as informative.

## Numbers

Values are shown as digits when representing the value of fields, and follows the IEEE Style Guide otherwise. For example, “set to 1” and “two packets”.

Also, specific usages:

* “0s”, “1s” and “2s”, **not** “zeros”, “ones” and “twos”
* “1s complement”, “2s complement”, “the register initialized with three 1s and two 0s”
  + **Not** “2s-complement” – i.e., no hyphen.
* “1-octet field” and “field of length one octet” are both correct

Bit positions and labels are numbered using an upper-case B, e.g., B12.

A reference to the value of a bit, e.g. b0, can be lowercase. In this case, “b0” represents the name of a variable.

Note that there is always a space between a number and its units (e.g., “20 MHz”). Long numbers have embedded spaces to group digits into threes (e.g., “65 635 octets”). Non-breaking spaces (ctrl+space) can be used to avoid unfortunate line breaks in this context.

Note that there is an exception for the MIB; the MIB does not have spaces in long numbers.

## Maths operators and relations

“x to y” is inclusive of the values of both x and y. Any use of “up to and including” should be avoided, because it casts doubt as to whether other uses of “x to y” are somehow not so well specified.

“up to y” is inclusive of the value y.

Int(), Floor() and Ceil() and their symbolic forms are defined in 1.5. These terms should be used in preference to other equivalent terms. There is no need for a “where” clause to repeat these definitions. The symbolic forms for Floor() and Ceil() should be used wherever possible.

In general, use the terms defined in 1.5, not synonyms. Do not repeat the definition of these terms elsewhere in the text.

Note, an Amendment does not exist in isolation, but in the context of its declared baseline. The terms in 1.5 can be used in an Amendment without further definition.

## Hyphenation

Most words created from a prefix and a word should not include a hyphen.

The editor closed up the following compound words (not enclosed in parentheses) in Std 802.11-2012. Those terms enclosed in parentheses have been added subsequently as fitting the pattern of terms that should not include a hyphen.

* nonadjacent
* nonauthorized
* nonbasic
* nonbufferable
* noncountry
* nondecreasing
* (nondifferential)
* nondynamic
* nonempty
* nonexcluded
* nonextended
* nonglobal
* nonidentity
* noninfrastructure
* noninteger
* noninvasive
* nonmember
* nonmesh
* nonmobile
* nonoperating
* nonoverlapping
* nonpeer
* nonperiodic
* nonpowersaving
* nonprotected
* nonreceipt
* nonreserved
* nonrobust
* nonstandard
* nontransmitted
* nontriggered
* nonvoice
* nonzero
* deaggregation
* deencapsulated
* descrambler
* dewhiten
* multiframe
* omnidirectional
* preassociat...
* preauthenticat...
* preexist...
* premodifier
* preplanning
* preprocess...
* preset
* preshared
* reactivation
* rearrange
* reassociat...
* redirect
* reestablish...
* rehabilitation
* reinitialize
* remap
* resent
* resubmit
* reused
* unadmitted

There are exceptions. The following are OK:

* non-initial
* non-monotonic
* non-negative
* non-null
* pre-robust
* fixed-length (hyphenated when before a noun)
* follow-up
* signal-to-noise
* STA-to-STA
* third-party
* variable-length (hyphenated when before a noun)
* vendor-specific
* multi-band
* non-duplicate
* co-channel

Note that IETF RFC xxx references do not use a hyphen in IEEE 802.11.

## References to SAP primitives

The name of a primitive should be viewed as an adjective. Add the word “primitive” to turn it into a noun phrase. So say: “The SME generated an MLME-NEIGHBORREPREQ.request primitive”.

Note there is no need to further decorate it, such as saying “an instance of the xyz primitive”.

## References to the contents of a field/subfield

The contents of a field or subfield are referenced in the following contexts:

1. When describing encodings (Frame Formats)
2. When describing normative behaviour for setting the value
3. When describing normative behaviour for testing or using the value

The following are examples of preferred usage:

1. When describing encodings (Frame Formats)
   1. The values of a field are often described in a table, e.g. “The <field> field specifies the <some description> as defined in <Table reference>.
   2. The table or Clause 8 text defining the (sub)field might include a description for enumerated values such as “set to 1 for no acknowledgement”, “set to 2 for normal acknowledgement”
2. When describing normative behaviour for setting the value, the verb “set” is used:
   1. “A STA that transmits an MPDU that is a retransmission shall set the Retry subfield of the Frame Control field of the transmitted MPDU to 1”
   2. “The STA shall transmit the MPDU with the Retry subfield of the Frame Control field of the transmitted MPDU set to 1”
3. When describing normative behaviour for testing or using the value
   1. A STA that received an MPDU in which the Retry subfield of the Frame Control field is 1 shall determine if the MPDU is a duplicate using the duplate cache.
   2. A STA that received an MPDU with the Retry subfield of the Frame Control field equal to 1 shall determine if the MPDU is a duplicate using the duplate cache.
   3. The use of “value of <field> field” is deprecated.[[1]](#footnote-1) So the following should not be used:
      1. A STA that received an MPDU with the value of the Retry subfield of the Frame Control field equal to 1 shall determine if the MPDU is a duplicate using the duplate cache.

## MIB attributes

MIB attributes shall be named according to the conventions in 11-09-0533-01-0arc-recomendation-re-mib-types-and-usage.ppt and 11-15-0355-13-0arc-mib-truethvalue-usage-pattern.docx.

References to MIB attributes should follow the patterns defined in 11-15-0355-13-0arc-mib-truethvalue-usage-pattern.doc.

There is no need to indicate that a cited variable is a MIB variable. It is obvious from its name.

So: “The STA sets the Event field to 1 when ~~the MIB attribute~~ dot11MgmtOptionEventsActivated is true, and sets it to 0 otherwise.” – i.e., delete any superfluous “the [MIB] (attribute|variable|table|parameter)”.

## Hanging Paragraphs and single child subclauses

A paragraph that occurs between a heading and children of that heading is called a hanging paragraph. It is not allowed because the scope of a reference to the heading is unclear – is it only the text that follows, or does it include all the children too?

Therefore, a subclause shall include either text, or child subclauses, but not both.

If even one sentence of “introduction” is necessary before child subclauses, this must go in its own child subclause. Such subclauses are typically headed “General” or “Introduction”. Don’t use “Introduction” if it contains any normative statements.

A subclause should not have a single child subclause; two or more should be present otherwise the child subclause heading is unnecessary since it is not separating material in the subclause. A single child subclause looks something like the following:

25.5.5 Dog

25.5.5.1 Dashund

25.5.6 Cat

## Abbreviations

Abbreviations may be defined for terms that are used frequently throughout the document. When an abbreviation has been defined, use it. (If you don’t the publication editor will probably replace most occurences in the text of the full term with the abbreviation).

But don’t create abbreviations for:

* Terms used only a handful of times
* Names of fields, structures, elements or frames

Do not include an abbreviation of the name of a field in the name of the field itself. e.g., a field labelled “Number of Taps (N\_taps)” is wrong.

Don’t create an abbreviation that includes the whole of a noun phrase – e.g., a XYZE being defined as an Xray Yankee Zulu element. This causes confusion because names are generally adjectives followed by a noun. TGmd is likely to move away from this usage.

## Format for code/pseudocode

Pseudo-code should be in courier font.

There are no other consistency requirements, but it is recommended to pick a style as close as possible to an existing example in the standard.

# Style applicable to specific Clauses

Clause numbering relates to IEEE Std 802.11-2016.

## Definitions (Clause 3)

Subclause 3.1 contains definitions that are consolidated by IEEE into a single publication of general definitions for terms used in IEEE standards (e.g. IEEE Standards Dictionary Online). Any definition that is considered “generally applicable in the industry” should be included in 3.1. All acronyms in subclause 3.1 have to be expanded on first use within each definition.

Many definitions that appear in 802.11 are more local, and have no meaning or significance outside of this particular document. Such definitions should be included in 3.2.

For example, the term “cipher suite” (a set of one or more algorithms designed to provide data confidentiality, data authenticity or integrity, and/or replay protection) is applicable outside of the 802.11 standard, and is included in 3.1. The term “Michael” (The message integrity code for the Temporal Key Integrity Protocol) is included in 3.2.

Abbreviations used in definitions should be spelled out in full on their first use in each definition. Example: “**access point (AP) path:** Path between two tunneled direct-link setup (TDLS) peer stations (STAs) via the AP with which the STAs are currently associated.”

Definitions should appear in alphabetical order. In addition, following rules apply:

* spaces sort earlier than letters or digits
* digits sort earlier than letters (but numbers sort in numeric order, e.g. 2 is before 10)
* ignore (treat as not present) everything that is not an ASCII alphanumeric or space, including hyphens
* ignore (treat as equivalent) case of letters
* ignore (treat as not present) stuff in parentheses.

## General Description (Clause 4)

Clause 4 provides a general description of the wireless system. It should be written in declarative, not normative, language.

## Frame formats (Clause 9)

Clause 9 is reserved for describing structure (apart from statements in 9.1).

Statements that describe the actions of a STA in order to determine a value for a field and any other behavioural specification should not be present in Clause 9.

This requires a bit of interpretation.

For example: “the Length field is set to the logarithm of the number of octets in the remainder of the frame” is acceptable. The act of calculating a logarithm is not considered to be behaviour.

But: “The Legacy Devices Present field is set to 1 when the STA receives a beacon that does not include an HT Operation element” is clearly a description of behaviour, and therefore not acceptable.

### Use of normative language in structure/field definitions

See 11-09-0433-01-000m-clause-7-normative-language.doc.

Normative language shall not be used to describe structure. I.e., you can say: “the structure consists of an 3-octet Length field followed by an Amplitude field” – although it is more typical to use tables and figures to define structure.

Normative language shall not be used for describing the encodings of fields. I.e., you can say: “the value 1 represents Measurement Enabled”, but cannot say: “the field shall be set to 1 to represent Measurement Enabled”.

The figures in Clause 9 are normative. Do not duplicate the length information given in figures in Clause 9 text. For example, “The Y field is x bits/octets in length" or "The Y field is an x-bit/octet field" are redundant. There is no value in duplicating the information.

## SAP Interfaces (Clause 6)

### Presence statements

Normative language shall not be used in “Presence” statements, such as occur in primitive parameter tables. These statements should, wherever possible, follow this template:

***The <name> <type of structure> is [optionally] present if <some condition>[; otherwise not present].***

### Consistency Requirements

The SAP interfaces should be reviewed for consistency. Unfortunately, most participants pay little attention to these interfaces, and they often become inconsistent with changes made elsewhere in the document.

The following consistencies should exist:

* The parameters in the .request should match (in some sense) the contents of a request frame.
* The parameters of a .indication should match the .request
* The parameters in the .response should match (in some sense) the contents of a response frame.
* The parameters of a .confirm should match the .response
* Any Reason Code values enumerated in a .request/.indication should
  + match, and
  + the names of these values should be present in the Reason Code table of the Frame Formats clause.
* Any Status Code values enumerated in a .response/.confirm should:
  + match, in the sense that the .confirm contains all the values of the .response, plus any locally-generated Status Code values,
  + the names of the values from the .response should be present in the Status Code table of the Frame Formats clause.
* The correct type for parameters of MAC addresses is “MAC address”, not “MACAddress” nor “MAC Address”.

### Primitive Patterns

A service’s primitives should fit one of the following patterns:

|  |  |
| --- | --- |
| **Type** | **Primitives** |
| 1 – Unconfirmed, local | .request |
| 2 – Confirmed, local | .request / .confirm |
| 3 – Unconfirmed, remote | .request / .indication |
| 4 – Confirmed, remove | .request / .indication / .response / .confirm |
| 5 – Event | .indication |

### Locally generated Status Codes

A .confirm primitive should not generally include locally generated status codes that represent:

* “You asked me to do too much”[[2]](#footnote-2)
* “You asked me to do something invalid”[[3]](#footnote-3)
* Transmission failure / success[[4]](#footnote-4)
* Response timeout (i.e., “did not receive an xyz response action frame in time”).

Transmission failure/success and response timeout may be present when specific protocol is defined for the SME that is dependent on these values.

## New top level clauses

An amendment that adds significant new PHY or MAC features, introduces these features in a new top level clause, e.g., “Very high throughput (VHT) PHY specification” or “High efficiency (HE) MAC specification”.

A new MAC specification should explicitly include inheritance statements. For example, a statement such as “An HE STA is a VHT STA.”

The use of top level clauses helps the reader place new features and modifications to existing features in their historical context by grouping features generationally. For example, MAC features introduced by the high efficiency enhancement amendment are largely collected in a top level clause “High efficiency (HE) MAC specification” rather than as subclauses or individual statements spread throughout clauses 10 and 11.

## Annexes

Annexes in a revision standard should be ordered (at some point during the balloting process) starting with Bibliography, then all normative Annexes, then all informative.

Annexes in an amendment should be added after all existing Annexes. An amendment should not attempt to modify the ordering of annexes in the baseline. If any reordering is required, it will be performed during a revision.

## Annex A – Bibliography

Annex A shall contain the bibliography. All references appearing in this bibliography shall be cited in the normative or informative text.

## Annex B – PICS

The 802.11 Standard shall include a PICS (Protocol Implementation Conformance Statement) proforma.

The level of detail to be included in the PICS is left to the discretion of the voters. Historically the PICS has identified major components of the wireless system, identified whether their implementation is mandatory or optional, and indicated any interaction between implementation requirements. Historically the PICS has not included an entry corresponding to each and every “shall” in the normative text.

Note that the PICS does not create requirements for conformance, it merely reflects what is stated in the body of the standard.

## Annex C – MIB

The 802.11 Standard shall include a MIB (Management Information Base).

Each MIB attribute shall be classified as capability, control, or status, as described in 11-09-0533-01-0arc-recomendation-re-mib-types-and-usage.ppt.

The definition of the MIB attributes with a TruthValue data type (“SYNTAX TruthValue”) shall follow the patterns defined in 11-15-0355-13-0arc-mib-truthvalue-usage-pattern.docx.

MIB authors should follow the recommendations in C.2. Each amendment should ensure that its MIB compiles, as described in C.1. This is a non-trivial task, as it requires rolling in the MIB from previous amendments to discover numbering and naming collisions. (This is a requirement of a WG802.11 MDR prior to sponsor ballot).

### Naming of MIB Variables

MIB Variables shall be named and described according to the conventions in 11-09-0533-01-0arc-recomendation-re-mib-types-and-usage.ppt.

### Description of MIB Variables

The DESCRIPTION of each MIB variable follows a standardized format:

Line #1: "This is a <type> variable" (either control, status, or capability)

Line #2: (control) "It is written by <writer>[ when <condition>]"

or (capability only)“Its value is determined by device capabilities.”

Line #3: (optionally) "The change takes effect <when>"

Follow these two/three lines with a blank line, then any further descriptive text

### Compliance requirements

This is a work in progress. However, it appears that we have repeatedly broken the IETF rules relating to modifying existing groups and compliance statements.

See document 11-11/0544, which states:

1. An amendment may not modify any existing group
2. An amendment must include each new variable in a new group
3. An amendment should not replace an existing group, e.g., no “dot11smtBase<n>” increments.
   1. Note it will be the job of the revision to sweep up any changes to these “fundamental” groups.
4. An amendment must provide a new module-compliance statement for each module it introduces new objects/group into.
   1. Each new group must be cited in at least one module-compliance statement.

It may be that the best way to procede is a middle-ground:

1. Allow the Amendments to break the IETF rules by editing existing groups and compliance statements
2. In a revision, define new groups and compliance statements for any that were modified by amendments to the previous revision, and deprecate the superseded ones.

Procedure of deprecating a MIB variable:

Step 1: Change the status of the MIB variable to “Deprecated”.

* Change its STATUS to “Deprecated”.
* Insert a new first line in the DESCRIPTION: “Deprecated as the related feature has been removed from the standard”.

Step 2: In the draft, find any reference to the variable in any GROUPs and re-instate this reference.

* Change the group’s STATUS to “Deprecated”.
* In the DESCRIPTION, insert a new first line: “Superseded by YYYY.” (Note that “YYYY” is the new GROUP name.)
* For each of the groups noted above, copy the group, set its status to “Current” and increment (or add) a number after the name of the group name (e.g. dot11SMTbase11 -> dot11SMTbase12).
* Make the corresponding changes (e.g. add or remove MIB varables) in the new group.

Step 3: For each reference to one of the noted groups from a compliance statement, update it to refer to the new group.

### Lexical requirements

The MIB needs to be compilable. The smtools compiler (see reference in C.2) requires 7-bit ASCII, or compilation will be aborted. Note the following:

* Greek or Unicode characters are not allowed. Spell out the units in full.
* Font effects, such as superscript, will be lost if the MIB is read in .txt format. For exponentiation, use “\*\*” (e.g., 2\*\*32).
* Long numbers in the MIB shall not have a space to group digits in thousands since this will affect compilation. Long numbers in a MIB object description should not have a space either; this is not consistent with style elsewhere in the standard (see 2.9), but the MIB is a standardalone entity and spaces here may lead to confusion.

## Annex G – Frame exchange sequences

An amendment that adds new frame exchange sequences shall update the normative Annex G.

It will not successfully exit the MDR unless it has satisfied this requirement.

# Amendment Editor Instruction

When using the “Insert” instruction, please identify the first sentence of the positioning paragraph.

For example, “Insert the following paragraph after the second paragraph (“The FT protocol provides ...”):”.

# Changes in the IEEE-SA Style Guide

This section summarises changes in the style guide.

## 2009 to 2012

* Scope and purpose statements no longer have to match exactly the PAR, they merely need to be within the scope of the statements in the PAR, **as determined by the (sponsor) balloting group**.
* The Standards Dictionary (IEEE 100) is now available as an online database:
  + <http://dictionary.ieee.org/dictionary_welcome.html>
  + Access is granted to WG officers and editors, and requires approval by our staff liaison.
  + The CD ROM can be ordered from Techstreet (<http://www.techstreet.com/cgi-bin/detail?product_id=1588912>, $213.00)
* Do not use words that imply any properties regarding safety (this shouldn’t be an issue for us).
* Example of “Hanging paragraph” added.
* New material describing use of italics in equations.
* Standards cited in a Bibliography can be either dated or undated, as appropriate.
* Chicago Manual of Style cited for various aspects of style.
* New guidelines and best practices for the creation and maintenance of IEEE standards terms and definitions.

## REVmc D1.0

Changes made based on the following comment resolutions:

* Optional fields in octet figures should show “0 or <n>”
* Clarifications on naming of frames (to remove unnecessary decoration and ensure consistent capitalization)
* Do not include local definitions of Element ID and Length fields in an element definition
* SIFS is already a noun
* twos complement -> 2s complement
* consistency of use of Int(), Floor(), Ceil() and their symbolic forms.

## REVmc D2.0

Changes made based on comment resolutions from REVmc D1.0:

* Definitions for PPDU and MPDU point to “frame” as an abbreviation of both, based on context.
* “Ack frame” preferred over “Ack MPDU”
* Bit numbering using “B<number>”
* Some requests and reports use initial caps – i.e., are a valid proper noun.

## REVmc D3.0 (MDR)

Include full names of frames, e.g. “ADDBA frame” is wrong.

”true” and “false” can be used in 802.1X state machine description

”should” permitted in informative Annexes and NOTES

”shall only” should be replaced with an alternative

“Shall ensure” is OK

Variables holding bit values can be called “b0”

## MIB design patterns

Reference 15/355r13

## REVmd D2.1 (MDR)

Repeated fields

Deprecate terms unicast and multicast

Figure titles: “Figure <number)—Name frame/field/element/etc format”

## REVmd D5.0

For proper names, all words in the name should be capitalized, including prepositions and conjunctions.

Arrow should not be used in the frame format figures, except where labelling parts of the structure or the MAC Header in the top-level frame format.

The figures in Clause 9 are normative, do not duplicate length information given in figures in Clause 9 text.

In the bit-aligned figure, the bit position should always start with B0.

Normative and Informative.

Amendment Editor Instruction

1. Note 802.11-2016 1.4 states: ‘If <x> represents a scalar field, scalar subfield, scalar parameter or scalar MIB attribute:

   — if “<x> is” is used in a context that relates to the testing or setting the value of “<x>” this usage is to

   be interpreted as though written “the value of <x> is”

   — “<x> indicate(s)” is to be interpreted as though written “the value of <x> indicate(s)”

   — “indicated by <x>” is to be interpreted as though written “indicated by the value of <x>”

   — “<x> that indicate” isto be interpreted as though written “<x> whose value indicates” ’ [↑](#footnote-ref-1)
2. Resource limits that are purely local are not relevant to the protocol and should not appear. This is different from resource limits that are negotiated on-the-air. These can be a proper subject of status codes. [↑](#footnote-ref-2)
3. See 11-11/0284 for a detailed rationale. But in brief, the reason is that both SME and MLME are in equal possession of knowledge about what is valid, from the point of view of operation of the protocol. The SME should not ask the MLME to do something that is invalid, and the MLME shouldn’t need to report such errors. [↑](#footnote-ref-3)
4. These may appear when there is an MLME protocol described that generates such a timeout, or an SME protocol described that makes use of such an event. [↑](#footnote-ref-4)